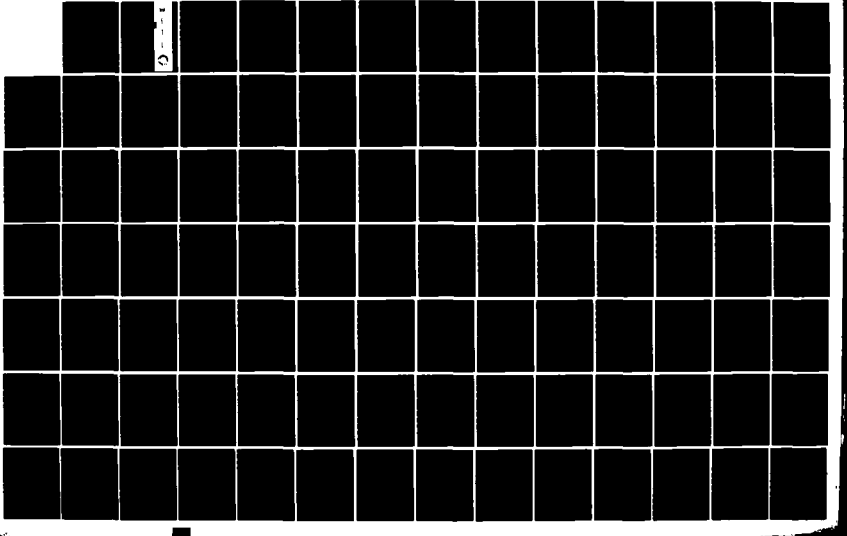
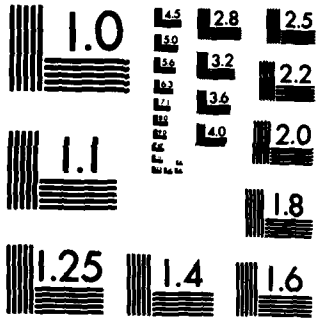
AD-A153 322 STUDY OF RASTER METAFILE FORMATS(U) BATTELLE COLUMBUS LABS OH M R TAYLOR ET AL. JAN 85 ETL-0363 DAAG29-81-D-0100 1/3

UNCLASSIFIED F/G 9/2 NL





ETL-0363

Study of raster metafile formats

Marshall R. Taylor
Peter N. French

Resources Planning Associates, Inc.
231 Langmuir Laboratory
Cornell Research Park
Ithaca, New York 14850

January 1985

Prepared for

U.S. ARMY CORPS OF ENGINEERS
ENGINEER TOPOGRAPHIC LABORATORIES
FORT BELVOIR, VIRGINIA 22060-5546

E T L

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ETL-0363	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) STUDY OF RASTER METAFILE FORMATS		5. TYPE OF REPORT & PERIOD COVERED Technical Report May 1984 - January 198[illegible] 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Marshall R. Taylor Peter N. French		8. CONTRACT OR GRANT NUMBER(s) DAAG29-81-D-0100
9. PERFORMING ORGANIZATION NAME AND ADDRESS Resources Planning Associates, Inc. 231 Langmuir Laboratory, Cornell Research Park Ithaca, NY 14850		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Engineer Topographic Laboratories Fort Belvoir, VA 22060-5546		12. REPORT DATE January 1985 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution is unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer Graphics Data Transfer Standards Metafiles Graphic Kernel System (GKS) Virtual Device Metafile (VDM) Raster Reformatting System (RRS)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report examines raster compatible, metafile systems in view of the needs of the Defense Mapping Agency Hydrographic-Topographic Center and the U.S. Army Engineer Topographic Laboratories. Background material is presented regarding the role of a graphic metafile standard in ETL/DMAHTC's cartographic applications. The issues relevant to the design of a raster compatible metafile system are examined. A recommendation is made to ETL/DMAHTC for adoption of the American National Standards Institute - Virtual Device Metafile proposed national standard.		

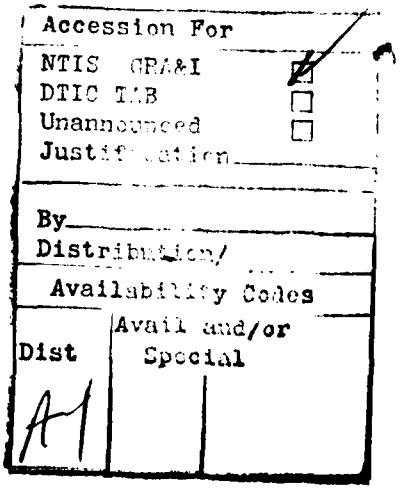
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

PREFACE

This document was prepared under contract DAAG29-81-D-0100 for the U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia 22060-5546, by Resources Planning Associates, Incorporated, 231 Langmuir Laboratory, Cornell Research Park, Ithaca, New York 14850. The Contracting Officer's Representative was Mr. Larry C. Cook.



ACKNOWLEDGMENTS

The participation of Battelle Columbus Laboratories as Contracting Agent under the Scientific Services Program is gratefully acknowledged. The review and consultation provided by Dr. John C. Dill was essential to the successful completion of this study. Special thanks are due to Mr. Larry Cook, Ms. Rose Holcheck, Mr. Richard Rosenthal and Mr. Dave Scott of ETL and to Mr. Art Noma of DMAHTC for their guidance of this study.

TABLE OF CONTENTS

ILLUSTRATIONS

TABLES

Introduction and Background

The last two decades have seen a tremendous growth in the availability and the sophistication of both digital cartographic products and cartographic information and display systems. Federal, State, and private organizations now believe that efficiency can be gained by the establishment of digital cartographic data standards. Necessary theoretical research and practical development are currently underway to establish data standards and to develop device-independent data exchange mechanisms (1). The National Research Council Panel to Review the report of the Federal Mapping Task Force on Mapping, Charting, Geodesy, and Surveying (2) recognized that these efforts were essential to the establishment of a useful national digital cartographic data base.

Work performed by the National Committee for Digital Cartographic Data Standards (NCDCDS), formed by the American Congress on Surveying and Mapping (3), provides a useful overview of the several diverse topics which must be examined during development of digital cartographic data standards. Figure 1, which is adapted from NCDCDS Report #4, illustrates the relationship between Digital Cartographic Data Standards and computer graphics and CAD/CAM. This report examines one element of that relationship: the design of device-independent graphic data exchange metafiles.

The current and future use of computer graphics to complete the missions of the U.S. Army Engineer Topographic

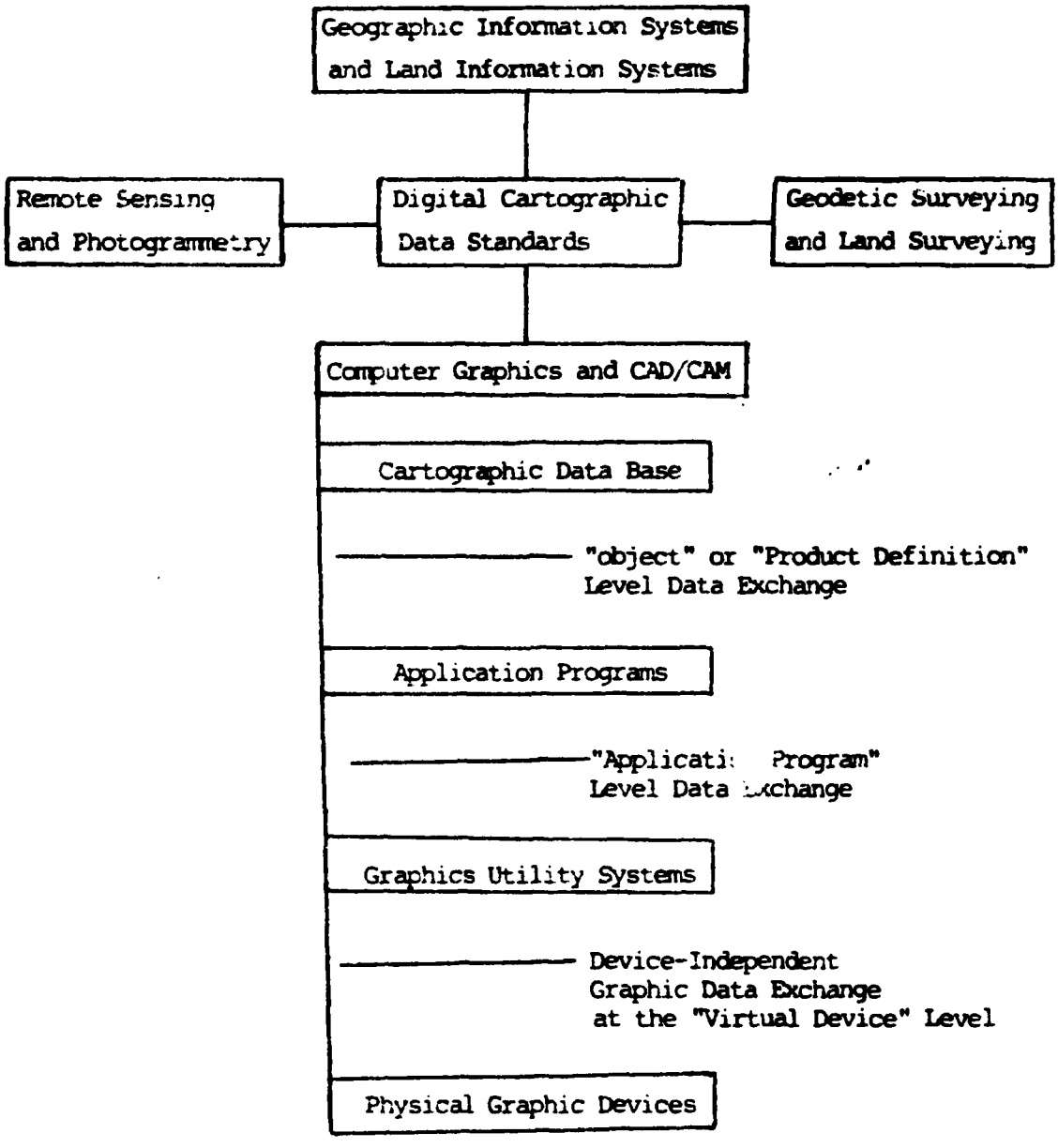


Figure 1. Overview of the diverse topics related to development of digital cartographic standards with emphasis on computer graphics and CAD/CAM.

Laboratories (ETL) and the Defense Mapping Agency Hydrographic/Topographic Center (DMAHTC) can be enhanced by the development of metafile formats compatible with raster graphic systems. This report summarizes a study of the needs of ETL and DMAHTC and contains recommendations for the development of a prototype metafile format. This metafile system would be incorporated into a raster compatible, device-independent graphics software system.

A metafile is a standard device-independent display-record format. A metafile, as defined in the contracting statement of work, has the following purposes:

- to provide a universal method for transferring graphic images between two computing sites,
- to provide an audit trail of image development,
- to provide a data source for hardcopies of images designed during interactive graphics sessions,
- to provide an archival medium,
- to assist in certification and verification of graphics data,
- to serve as an interface standard for intelligent peripherals.

Additionally, the statement of work declared that a proposed metafile design for use with raster graphics systems must be capable of handling scanned images, text, and synthetically generated graphics.

A well designed metafile system must meet several additional criteria. First, the system must be functionally declared so that it can serve as a standard graphics data interface for software developers. This characteristic is

important for reducing the cost of training software developers and for aiding the transportability of software between installations. Second, the metafile system should provide standard escape mechanisms which allow non-standard data to be recorded, and which allow access to non-standard graphic device capabilities. As will be seen later in this report, a standard escape mechanism should be a requirement for any proposal for an ETL/DMAHTC raster metafile system; it will play an important role in the proposal later described. Finally, care must be taken to provide for the future functional extension of the metafile system.

The authors used the following methods to study the applicability of proposed metafile systems to ETL/DMAHTC operations:

1) discussions with ETL and DMAHTC staff provided data, beyond that contained in the statement of work, concerning the needs of ETL and DMAHTC, and concerning other graphic and programming standards under consideration by those organizations,

2) the production environment of DMAHTC was examined by review of the "Baseline Systems Description for the Cartographic Systems Integration and Upgrade-Interim Technical Report #1", prepared by ZYCOR, Incorporated in January 1984 (4),

3) the current technical literature including activities and proposals of the American National Standards Institute X3H3 (ANSI-X3H3) Technical Committee (5), was thoroughly reviewed,

4) the "Raster Reformatting System Design Plan" submitted to the Rome Air Development Center by Synectics Corporation in September 1983 (6) was reviewed, and

5) discussions were held with individuals who are knowledgeable in the field of computer graphics and metafile design and who would be aware of emerging

raster metafile proposals likely to receive significant attention.

The purpose of the remainder of this report is to summarize the findings and to present the authors' conclusions and recommendations. The next section discusses some of the most relevant issues which must be addressed in the design of a metafile useful for ETL and DMAHTC. Following that discussion is a brief review of the American National Standards Institute's Virtual Device Metafile (ANSI-VDM) proposed standard and a recommendation regarding the extension of that standard for use as a raster metafile standard. The next section addresses several key elements of the Synectics report and discusses how elements of the design described in that report could serve as a basis for the proposed VDM extension. Finally, the authors' conclusions and recommendations are presented, with the appendices providing bibliographic material and technical documentation.

Issues Relevant to the Design of a Raster Metafile

Both private and public organizations have developed proposals or standards for the specification of a graphic metafile system. Among these are:

1) the proposals of the Association for Computing Machinery Graphic Standards Planning Committee (ACM-GSPC) (7),

2) the American National Standard Institute (ANSI) standard for transporting Computer-Aided Design (CAD) and Computer Aided Manufacturing (CAM) data through the Initial Graphics Exchange Specification (ANSI-IGES) (8),

3) the North American Presentation Level Protocol Syntax (NAPLPS) (9), and

4) the Virtual Device Metafile proposals of the ANSI-X3H3 technical committee (5).

Not all of the systems listed above are designed to meet the same needs, and their relationships with other adopted or proposed graphic standards must be understood by those responsible for implementing a graphic metafile system. For example, ANSI-IGES is concerned with the transfer of all product definition data (geometric and nongeometric) between CAD/CAM systems and installations. The function of the VDM proposed standard is the generation and transfer of sufficient device independent information for a graphic representation to be presented on a wide variety of graphic output devices. **To perhaps oversimplify the distinction, ANSI-IGES is designed to allow the transportability of the full definition of objects, whereas ANSI-VDM contains only that information necessary to present a picture of those objects.** In

most applications, data transferred by ANSI-IGES allow the digital representation of an object to be further processed and analyzed. In general, ANSI-VDM compatible files only provide a device-independent means to present a picture, not to continue manipulation and processing of the digital representation of the illustrated object.

Figure 2 illustrates the various levels on which the most widely discussed graphics standards are designed to operate. At the top of Figure 2 is an "object database". This element of the illustration represents computer descriptions of "objects" which can be defined in 2 or more dimensional space. For example, digital descriptions of transportation networks, buildings, and other physical features which can be graphically depicted may be a part of an object database. The major purpose of ANSI-IGES is to serve as an interface between the database and application programs which need to access those data. In addition, ANSI-IGES establishes a standard under which CAD/CAM installations may exchange data.

The interface between application programs and a particular installation's graphics utility system is handled by such proposed standards as ANSI-Graphical Kernel System (ANSI-GKS) and ANSI-Programmers Hierarchical Interface to Graphics Systems (ANSI-PHIGS) (10). Both of these proposed standards allow application program development to proceed without concern for the specifics of an installation's graphics utility system. Similarly, ANSI-GKS and ANSI-PHIGS

allow the development of a local utility system before specifying the requirement of the application programs to be installed (11). As with ANSI-IGES, industry adoption of GKS and PHIGS facilitates the portability of application programs between differently configured installations.

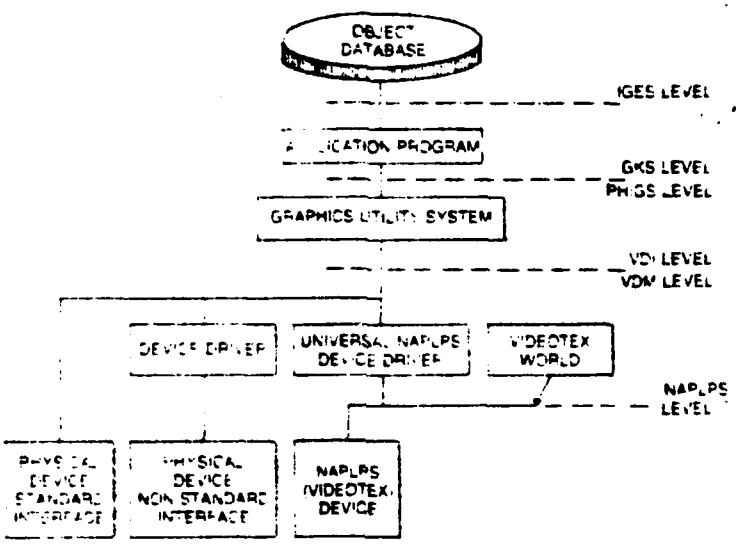


Figure 2. An illustration of the various levels on which graphic standards are designed to operate and the levels of some of the current and proposed standards. Source: National Computer Graphics Association.

The last level illustrated in Figure 2 which will be discussed in this report is labelled as the VDI/VDM level. These labels correspond to the ANSI-Virtual Device Interface (ANSI-VDI) and the ANSI-Virtual Device Metafile (ANSI-VDM) proposed standards. VDM and VDI were designed by the ANSI committees to standardize the interface between graphics

software and graphics devices. As illustrated in Figure 2, a VDI compatible physical device would not require any additional software to interface to a VDI compatible graphics utility system. Physical devices that were not VDI compatible still would require specific device drivers, which currently is the industry norm. As proposed by ANSI, VDM specifies the format of a device-independent metafile used to record and transfer device-independent pictures.

Evident from Figure 2 is the fact that the development of a standard at one level must take into account the development of standards at the next lower level, at least. For example, the development of the proposed GKS 2-D graphic programming standard was, in many respects, accompanied by the development of the VDM 2-D graphics metafile standard.

Discussion with ETL/DMAHTC staff and review of the DMAHTC production equipment and procedures indicate that an immediate need exists, both at the application program - graphics utility system level and at the interface between the graphic utility system(s) and physical devices, for a raster metafile system. Throughout the remainder of this report those products produced by DMAHTC which require metafile support are divided into two broad classes. The first class consists of cartographic products based on the collection; interpretation; and analysis of geodetic data, topographic data, hydrographic data, geographic names data, and other physical and cultural information. The second class of products consists of grid based digital elevation models and similarly represented surfaces. **While it is essential**

for an object definition metafile standard at the IGES level to allow full 3-D object specification, the authors have concluded that there is not an immediate need for a full 3-D graphic metafile implementation within ETL/DMAHTC. The relevant ANSI committees are currently studying 3-D extensions to ANSI-GKS and ANSI-VDM. Adoption of the current ANSI proposals will provide ETL/DMAHTC with the best opportunity to adapt to the future 3-D standards.

The graphic metafile proposal identified by the authors as likely to be most widely implemented in organizations with activities similar to ETL/DMAHTC is the ANSI-X3H3 Virtual Device Metafile (VDM) proposed standard. A copy of the draft standard and public comments on that draft are included as Appendices C and D of this report. One of the reasons that VDM appears to best fit ETL/DMAHTC's needs is that it currently provides a rudimentary pathway to the inclusion of raster graphic data. Although the current specification of VDM does not include sufficient elements to serve as a complete raster metafile system, the authors believe that the adoption of VDM and the development of appropriate extensions will provide ETL/DMAHTC with the best available mechanism for a single metafile system encompassing both vector and raster graphics.

One of the original criteria for the design of the VDM proposal specified that VDM was to serve as a graphic standard across all types of graphic devices. Since the development of [illegible] standards inevitably lag behind

the development of technology, several criticisms of VDM have focused on it's failure to incorporate the full functionality of current raster display devices. **If these criticisms are adequately addressed by future extensions to ANSI-VDM, then the apparent need to have a raster metafile system separate from a general graphics metafile system will no longer exist.** The authors recommend that ETL/DMAHTC adopt this approach as an alternative to the development of a raster metafile standard separate from the standards used for other graphic data. ETL/DMAHTC should realize significant benefits by adopting this emerging industry standard rather than developing their own, distinctly raster, metafile system.

The above recommendation is further reinforced by the interest of the ANSI-X3H3 committee in providing a mechanism for adopting "registered extensions" to VDM. This procedure allows interested parties to submit useful VDM extensions for consideration by the committee. If the committee views an extension as meeting the criteria of the VDM standard, it will be released for public comment and considered for adoption as a "registered extension" (12).

Should ETL/DMAHTC participate in the "registered extension" process, two significant benefits could be realized both by ETL/DMAHTC as well as by the rest of the computer graphics industry. First, ETL/DMAHTC would provide a technology development and transfer service which would increase the benefits others obtain by the adoption of VDM. Perhaps a more important benefit of a registered extension designed

to ETL/DMAHTC specifications is that it would help insure that greater amounts of graphic data produced outside of ETL/DMAHTC operations would be readily accessible for their use.

The use and implementation of VDM is illustrated in Figures 3 and 4. In one use, illustrated by Figure 3, an application program using a device-independent graphics package, such as ANSI-GKS makes use of a metafile generator to produce VDM compatible output files. These files contain interim or final graphic data which are preserved by the VDM system as device-independent pictures.

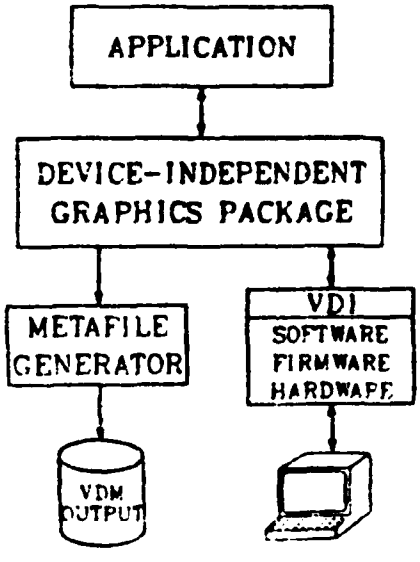


Figure 3. Illustration of the relationship of VDM to a device-independent graphics package and an application program. Source: ANSI-X3H3, #X3.122-198x.

Transfer of VDM compatible device-independent pictures to another graphic system is illustrated by Figure 4. In that figure, a set of VDM output files is processed by software called a metafile interpreter and, with the use of a Virtual Device Interface, may be displayed on any graphic device. Incompatibilities between the system on which the pictures were originally produced and the targeted display device are handled at the VDI/device driver interface.

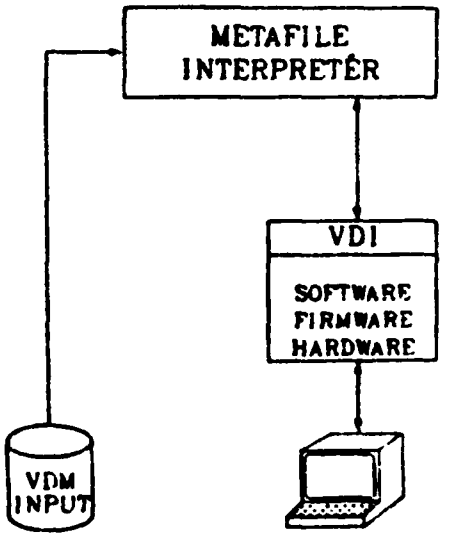
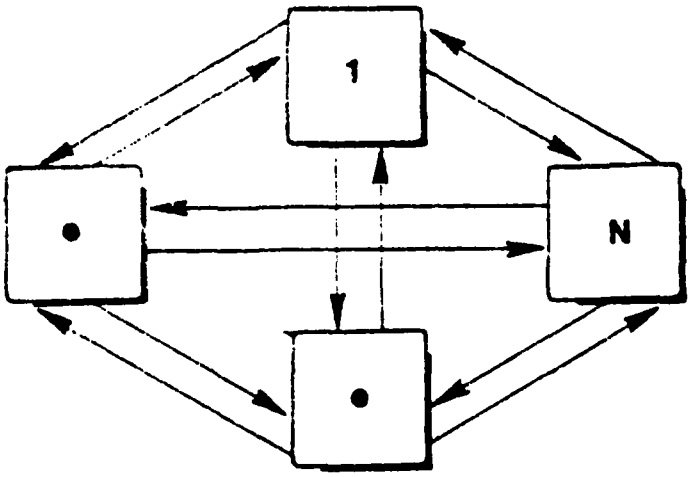


Figure 4. Illustration of the relationships between a VDM compatible picture file, a metafile interpreter, and a targeted graphic display system. Source: ANSI-X3H3, #X3.122 - 198x.

Note that figure 4 does not show VDM output files being used as input to an application program. As previously discussed, VDM is a picture metafile standard as opposed to an object description standard such as IGES. In general, an application program receiving VDM input does not have all the information required to continue definition and manipulation of the original objects(s) (13). This distinction between product definition and graphic metafile systems should, however, have little impact upon the use of VDM by ETL/DMAHTC. ETL/DMAHTC currently have standard cartographic data formats and cartographic data base management systems, and their primary uses of VDM will be for product verification, sequential pictoral processing, and final product archival and shipment in digital format. For example, an important need is to be able to review interim cartographic products on any of a large number of graphic devices. This interim product may have been produced on any of a large number of devices using several different application programs. The graphic data which are displayable on device A need to be displayed on device B. Using the picture produced by B, which may be a hardcopy device, the graphic data will be checked and verified. Detected errors will be noted, and subsequent corrections will be made on the original machine with an appropriate application package. Thus, the graphic is reviewed as a representation of the cartographic data. It is not necessary for the graphic metafile standard to maintain all of the relevant cartographic data.

Another example illustrates the use of a graphic metafile standard within the direct DMAHTC production process. Often various workstations, which may have been supplied as turnkey systems, are designated to perform only certain functions. Thus, a DMAHTC product may proceed through a "pipeline" which sequentially moves the product through various workstations. Each workstation produces a portion of the final product and defines cartographic data which must be preserved or passed to other workstations following ETL/DMAHTC standard procedures for processing cartographic data. It is desirable to have graphic data produced on a given workstation displayable on any number of other, significantly different workstations. Previously, as noted in the report by Synectics Corporation (6), this has been accomplished by developing and maintaining n(n-1) programs designed to change the graphic formats of data from one system to the other, where n is the total number of different types of workstations. The adoption of a graphic metafile will reduce this problem to one of developing and maintaining n metafile generators and interpreters. This advantage is well illustrated by Figure 5 which is adapted from the Synectics report (6). At the top of Figure 5 the case of no adopted metafile system is illustrated. The point made is that without a standard metafile a total of n(n-1) conversion programs must be developed and maintained if all n graphic configurations are going to be able to communicate. In addition, changing or acquiring a new system has an impact on the communications

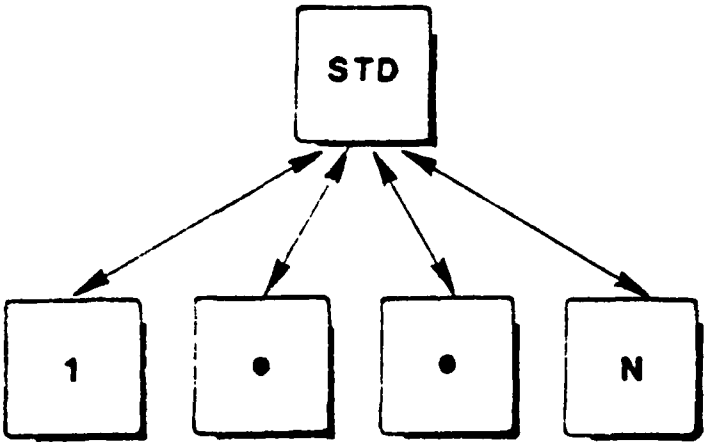
NO METAFILE STANDARD



* N(N-1) CONVERSION PROGRAMS

* N SYSTEMS IMPACTED BY SINGLE SYSTEM FORMAT CHANGE

ADOPTED METAFILE STANDARD



* 2N CONVERSION PROGRAMS

* 1 SYSTEM IMPACTED BY INTERNAL FORMAT CHANGE

Figure 5. Intersystem communications schemes before and after adoption of a standard metafile format.

software of all n graphic systems. This "no metafile" scenario can be contrasted with the "adopted metafile" case by examining the lower portion of Figure 5. Given an adopted metafile format, it is only necessary to insure that each system adapt to that metafile. With an adopted metafile, the effects of system changes are easily localized without producing any impact on associated graphic systems. Further, when additional workstations are acquired, it will only be necessary to specify that the workstation contain a metafile generator/interpreter in order to gain communications with all other workstations.

An important point not made by Figure 5 is that **adoption of an industry wide metafile standard (such as VDM) produces a communications path to many graphics systems external to ETL/DMAHTC.** Also, the authors believe that VDM compatibility is what will most likely be offered and supported by hardware vendors in the immediate future. **Such a commitment by hardware vendors will reduce the necessity of ETL/DMAHTC producing metafile generators/interpreters for future graphic system acquisitions.** Finally, the use of VDM will provide a standard archival format for ETL/DMAHTC products and will place those products in a digital format which is compatible with the emerging industry standard. Since close contact has been maintained between ANSI-X3H3 and the relevant International Standards Organization Committees (ISO TC97/SC18 and ISO TC97/SC2/WG8) it is also likely that revisions to VDM will be made in such a manner as to enhance the international exchange of graphic data and software.

Specifics of a VDM Based Raster Metafile Implementation

This section assumes some familiarity with at least two additional documents. The first is the ANSI-VDM public review document which has been reproduced as Appendix C of this report. The second is the "Raster Reformatting System (RRS) - Design Plan" prepared by Synectics Corporation in September 1983. That document has not been made a part of this report. Readers are encouraged to have a copy of the Synectics report available for reference as they review this section.

The RRS schema captures a portion of the functionality and benefits which accrue from the adoption of a standard metafile. However, RRS fails to meet the criteria of a metafile design in several ways. First, there was no attempt to develop a specification which would encourage application outside the ETL/DMAHTC/RADC environments. Also, there was apparently little consideration given to likely changes in hardware capabilities and provision of a planned method for increasing the functionality of the RRS specification. In addition, the Synectics report represents RRS as software which would be implemented on "gateway" computer systems. This approach, although reasonable for the short run, should not be an ultimate solution.

An additional concern that the authors have about the RRS schema is that it does not provide a pathway for the eventual merging of vector and raster data into a single metafile. **The adoption of a metafile standard for only**

raster data runs opposite to the prevailing movement of the graphics industry. The authors believe that ETL/DMAHTC adoption of an exclusively raster metafile standard would retard the development of graphic metafile standards.

The proposed VDM standard represents a well designed graphic metafile system. However, VDM has a limited set of functional specifications which deal with those capabilities of raster devices which do not have direct counterparts in vector devices. For many applications, such as working with satellite imagery and other scanned data, ETL/DMAHTC will require extensions to VDM to include the required functional specifications. For example, VDM makes only limited provisions for the use of data preserving compaction techniques, such as run-length-encoding, within a standard VDM file. No provisions are made for entropy reduction encoding methods. However, within the specifications of VDM, non-standard graphic data and the use of special device capabilities may be included in a VDM file with the use of a standardized escape mechanism. In addition, non-graphic data may be included with the use of an application data flag. Through these mechanisms, most of the encoding and data header recommendations contained in the RRS report can be implemented within VDM files. As previously mentioned, direct extensions to VDM could also be developed by ETL/DMAHTC. **Such extensions would be very useful to the field of digital cartography and, in particular, to the efforts of USGS and the National Committee for Digital Cartographic Standards.**

The most important raster-oriented functional element of VDM is the CELL ARRAY. A description of the CELL ARRAY is presented in S 5.5.9, page 76 of the appended VDM report. Figure 6 will be used to explain the function of a CELL ARRAY. Three corner points, labelled P,Q,R in Figure 6, define a rectangular area specified in a device independent coordinate system know as Virtual Device Coordinates. This area is subdivided into dx*dy contiguous rectangles which partition the rectangular space into identical cells spanning P-R and R-Q. The CELL ARRAY element allows the points P,Q,R and the cell size dx,dy to be specified, and allows data which are to be used to fill those cells to be encoded. The first encoded array element is mapped to the cell positioned at corner P, and subsequent array elements are mapped in rows running from P to R in row order from R to Q.

The precision of the specification of the array data is determined by other elements of the VDM specification (see COLOR PRECISION, S 5.3.7, page 46 and COLOR INDEX PRECISION, S 5.2.8, page 46). Whether the encoded values are direct color values specified in RGB color space or are indices into a previously specified color lookup table is selected by the COLOR SELECTION MODE specifier (S4.4.2, page 17). An elementary run-length-coding capability is described in S 7.6.1, and may be compared with bit-stream-coding described in S 7.6.2.

At this point it is instructive to compare the capabilities of VDM with the pixel encoding scheme proposed by Synectics. Reference should be made to section 3.4 of the

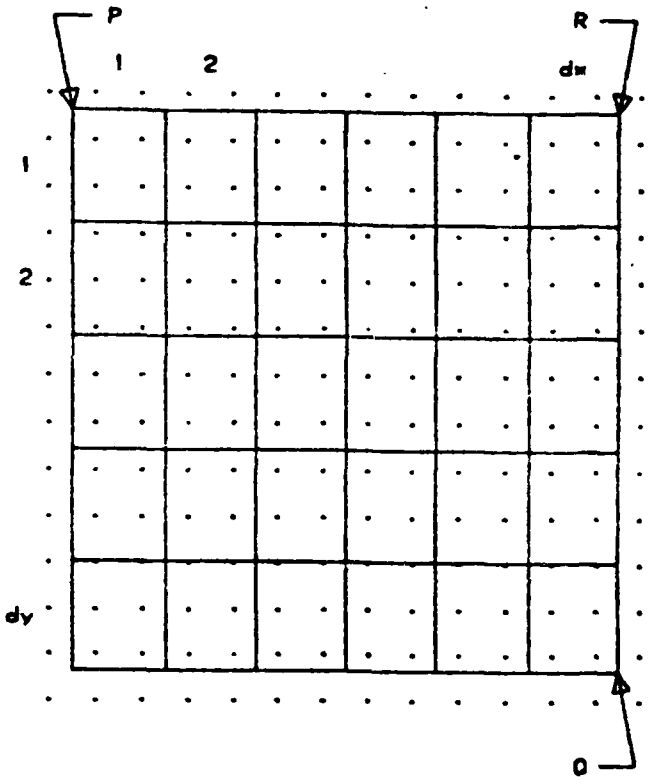


Figure 6. Illustration of the specification of a CELL ARRAY. A dx by dy rectangle is mapped onto the display surface. Lines indicate cell array locations. Dots indicate pixels.
Source: ANSI X3H3, #X3.122 - 198x

RRS report. First, note that the Synectics proposal specifically supports three types of raster data for encoding:

1) Bilevel images (black & white)
2) Gray scale images (8 bit maximum), and
3) 8 bit color (either direct or indexed).

For each of these three raster data types Synectics has proposed an efficient coding scheme both in terms of storage requirements and in apparent decoding computations. The ANSI-VDM specifications supports more flexible, but perhaps less efficient, coding schemes. The major difference between the two is that the ANSI proposal allows color information to be encoded to any level of precision, as denoted by the COLOR PRECISION and COLOR INDEX PRECISION specifiers.

Table 1 provides an overview of the storage required to encode simple images which fit into each of the three categories permitted by RRS. The elements of Table 1 are the storage (in kilobytes) required to store 1760x1024 raster images under RRS and VDM schemas. The first two images are black and white binary images. Image 1 is a simple checkerboard design with alternating black and white pixels. Image 2 is constant black or white. Image 3 is an 8 bit gray scale image of constant gray level. Images 4 and 5 are full color images consisting of 3 separate 8 bit channels for the red, green, and blue color components. Image 4 is a 3 color checkerboard where no adjacent pixels have exactly the same value in any of the three channels. Image 5 is also a 3 channel image, except that the color of each channel's image is constant throughout.

The reader will note from Table 1 that one of the VDM schemes is always the most storage efficient alternative for these five images; however, the authors do not present this as a general conclusion. In addition the VDM encoding mechanisms are significantly more computationally demanding. The authors believe that the encoding flexibility of VDM warrants acceptance of its additional computational demand. If computation time is a major concern to ETL/DMAHTC, **the authors suggest that ETL/DMAHTC design and perform benchmark comparisons between RRS and VDM for selected typical images.**

Table 1

Comparison of storage requirements for 1760 x 1024 raster images under RRS and VDM pixel encoding schemes

	STORAGE REQUIRED (kilobytes)		
		VDM	
	RRS	Bit Stream	Run Length Encoded
Binary Images			
1) checkerboard	1760	251.4	7040
2) constant black	13.8	251.4	0.2
8 bit gray scale			
3) constant gray	6.9	2346.7	0.2
24 bit color images			
4) 3 color checker-board	15840	7040	10560
5) constant color	.3	7040	.2

In the evaluation of the pixel encoding schemes, and of the differences between VDM and RRS in general, ETL/DMAHTC should not ignore the availability of the VDM ESCAPE element, the APPLICATION DATA element, and the MESSAGE element. The authors recommend the use of these elements to implement critical parts of the RRS design in a manner that would maintain VDM compatibility.

The VDM ESCAPE is briefly described in S 5.7. It's purpose is to allow use of device capabilities not specified by the standard while preserving the general transportability of VDM files.

The ESCAPE element has an associated integer function identifier which is used to specify the particular escape function necessary for processing its data list. The values of valid function identifiers are determined by prior agreement between metafile generators and interpreters. The ESCAPE element specifically applies to graphic data.

Data which are non-graphic and have no direct effect on targeted display devices may be included in the VDM APPLICATION DATA element described in S 5.8.2. Using the same general approach as the ESCAPE element, the contents of the APPLICATION DATA list may include information which instructs the metafile generator and interpreter to supplement the standard VDM data in an application-dependent way.

The VDM MESSAGE element (S 5.8) is used to encode a string of characters which is used to send messages to operators during VDM interpretation. These data have no effect

on the normal graphical output. One of the parameters of the MESSAGE element is a flag which indicates whether or not operator intervention is required. The VDM interpreter must determine whether it is permissible to continue or whether interpretation must pause and wait for an operator response.

Through the appropriate use of the ESCAPE, APPLICATION DATA, and MESSAGE elements of VDM, all the functionality of the RRS design may be achieved. Some efficiencies in storage requirements and computational effort may be lost. **The authors recommend that VDM be adopted as the raster metafile standard and that ETL/DMAHTC begin adapting the critical elements of the RRS design to the specifications of VDM. While this effort is underway the areas in which VDM functionality must be extended for ETL/DMAHTC applications should be concretely documented and the development of extension proposals considered.**

Conclusions and Recommendations

The authors offer the following conclusions and recommendations:

1) ETL/DMAHTC should focus attention on the development of a single, device-independent metafile standard instead of considering the development of a raster metafile format separate and distinct from their other graphic data.

2) ANSI-VDM is the current metafile proposal at the device-interface level which is most likely to be offered and supported by graphic hardware vendors in the immediate future.

3) ANSI-VDM should be adopted as an ETL/DMAHTC graphic, device-level metafile standard and as a framework on which to build required raster capabilities which are not currently included in the VDM specification.

4) ETL/DMAHTC should examine the data elements incorporated into Synectics' Raster Reformatting System (RRS), and should develop guidelines for embedding the essential data elements of that system into VDM compatible files.

5) Required extensions to the current VDM specifications should be developed within the criteria of VDM and should be submitted for consideration as "registered extensions."

6) The development and implementation of a full 3-D metafile standard is not needed to support DMAHTC's current production needs. Current ANSI efforts to develop 3-D extensions to GKS and VDM should be monitored by ETL/DMAHTC.

7) ETL/DMAHTC should explore the possibility of establishing a representative on the ANSI-X3H3 Technical Committee and/or provide input to the efforts of that committee through the current U.S. Army representatives.

8) ETL/DMAHTC should continue their support of the National Committee for Digital Cartographic Data Standards and should submit this report, or extractions thereof, for appropriate committee review.

9) ETL/DMAHTC should initiate a study to develop implementation guidelines and specifications for the adoption and implementation of the current specifications of ANSI-VDM.

10) ETL/DMAHTC should initiate a study which examines IGES and similar metafiles and which establishes a framework under which a cartographic data metafile standard may be developed. This study should be performed in conjunction with, or as a supplement to, the efforts of the National Committee for Digital Cartographic Data Standards.

REFERENCES AND NOTES

1) The U.S. Bureau of Standards asked the U.S. Geological Survey to lead this effort and specifically asked USGS to develop standards for earth science information systems (see: USGS Digital Cartographic Data Standards: Overview and USGS Activities, Geological Survey Circular 895-A and Issues in Digital Cartographic Data Standards, Report #1, Harold Moellering, Editor, American Congress on Surveying and Mapping, July 1982).

2) National Research Council, 1980, Need for a Multipurpose Cadastre, Washington: National Academy Press.
National Research Council, 1981, Federal Surveying and Mapping: An Organizational Review, Washington: National Academy Press.

3) Moellering, Editor, 1984, Issues in Digital Cartographic Data Standards, Report #4: Digital Cartographic Data Standards: Examining the Alternatives, National Committee for Digital Cartographic Data Standards, American Congress on Surveying and Mapping.

4) Baseline Systems Description for the Cartographic Systems Integration and Upgrade - Interim Technical Report #1, Report No. 096-68-06. submitted to the U.S. Army Engineer Topographic Laboratory under contract DAAK7D-83-C-0078, January 9, 1984.

5) Draft Proposed American National Standard - Virtual Device Metafile, American National Standards Technical Committee X3H3 report #X3.122, Project #347, December, 1983. (This document and summaries of comments received during its recent public review are reproduced in appendices C and D.)
See also:
Draft Proposed American National Standard - Graphical Kernel System, American National Standards Technical Committee X3H3 report #X3.124, Projects #268, #362, January, 1984.

6) Raster Reformatting System (RRS) Design Plan, submitted to Rome Air Development Center, Griffiss Air Force Base, New York pursuant to contract F30602-83-C0068 by Synectics Corporation, September 1983.

7) Status Report of the Graphic Standards Planning Committee, Computer Graphics, Vol. 13, Number 3, August 1979, ACM Order No. 428791.

8) "IGES - Initial Graphics Exchange Specification", by Bradford Smith in NCGA '81 Conference Proceedings, 1981. "Market Considerations Hinder Search for Software Standards", by Carl Warren in Mini-Micro Systems, September 1984.

9) "Computer Graphics and Videotex", Computer Graphics World, October 1982.
"NAPLPS: More Than Just Videotex, "Computer Graphics World, February 1984.

10) American National Standards Institute, Programmers Hierarchical Interface to Graphics, ANSI-X3H3, 1984.

11) For those readers familiar with the specifications of GKS and PHIGS their positions in Figure 1 do not imply their relative level of sophistication. They both serve as programming interfaces at the level shown, even though PHIGS specifies a hierarchical and interactive capacity.

12) This proposal was explained in discussions with Dr. Peter R. Bono, current chairman of the ANSI Committee X3H3 on computer graphics. See comments by Dr. Bono in Computer Graphics Today, Vol. 1, No. 1.

13) An illustration of an exception is indicated in Figure 7, shown on the next page. That figure illustrates the use of a GKS compatible program to both read and cause the preparation of VDM metafiles. The lack of product definition information in the graphic VDM metafile limits the operation of the application program. A typical use of the illustrated operation would be to extract pictorial data from a metafile, to use the application program to append additional graphics, and to display and/or record as a VDM file the merged graphic.

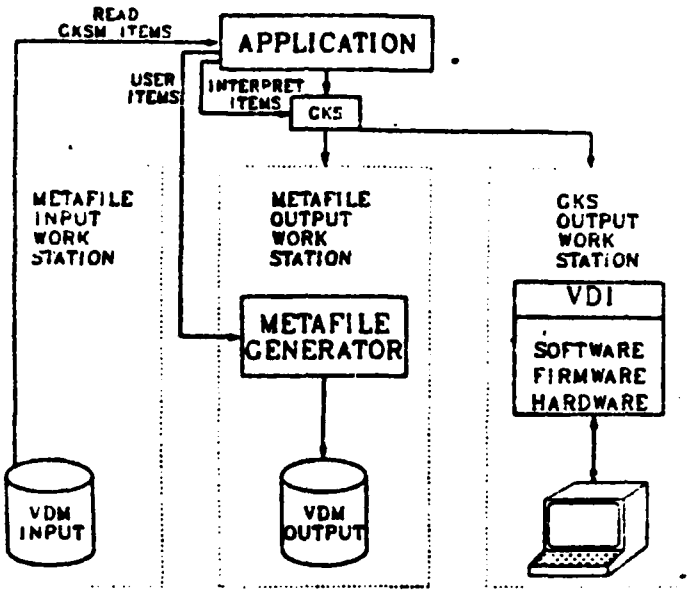


Figure 7. Illustration of the relationships between ANSI-VDM, ANSI-GKS, an application program, and a targeted output device.

ADDITIONAL REFERENCES

"Design Specification Extension is Graphics Milestone," Government Computer News, October 1984.

"Guest Viewpoint - UNIX and GKS Standards Create Opportunities for Graphics", by William Elmore in Mini-Micro Systems, September 1984.

"Integrating Videotex With the Personal Computer," by John Davidson in Computer Graphics World, May 1984.

"The GKS Advantage," by Clinton Waggoner in Computer Graphics World, October 1984.

"GKS Graphics Standard Replaces Core System for International Use," Computer Technology Review, Summer 1983.

"GKS Style Application Coding," by Dr. Barry Shepherd in NCGA '83 Conference Proceedings, 1983.

"A Terminal Protocol for Graphics Application," by Dr. W. Kelly in NCGA '81 Conference Proceedings, 1981.

"An Approach to Reducing the Barriers to Information Exchange," by Sam Steppel and Richard Berman in NCGA '81 Conference Proceedings, 1981.

"Data Communication Support for Computer Graphics," by Dr. W. Lanauer and Dr. O. Mowafi, in NCGA '81 Conference Proceeding, 1981.

Series of papers presented in NCGA '81 Conference Proceedings in the session on Device Independent Software chaired by Dr. Peter Bono.

"Image-Processing Software Portability Using a Conceptual Frame Store," R.J. Stevens and S.T. Alexander, Pattern Recognition Letters (1), 1983.

"An Object-Oriented Graphical Kernel System," by Thomas Lubinski and Ingeborg Hutzel, Computer Graphics World, July 1984.

"Graphics Standards Nearing National Adoption," Government Computer News, October 1984.

Series of articles on Software and Standards in the February 1984 issue of Computer Graphics World.

LIST OF ABBREVIATIONS AND ACRONYMS

ACM	Association for Computing Machinery
ANSI	American National Standard Institute
ANSI-X3H3	ANSI-Technical Committee #X3H3
CAD/CAM	Computer Aided Design/Computer Aided Manufacturing
DMAHTC	Defense Mapping Agency Hydrographic/Topographic Center
ETL	U.S. Army Engineer Topographic Laboratories
GKS	Graphical Kernel System
GSPC	Graphic Standards Planning Committee
IGES	Initial Graphics Exchange Specification
ISO	International Standards Organization
NAPLPS	North American Presentation Level Protocol Syntax
NCDCDS	National Committee for Digital Cartographic Data Standards
PHIGS	Programmers Hierarchical Interface to Graphics Systems
RADC	Rome Air Development Center
RRS	Raster Reformatting System
VDI	Virtual Device Interface
VDM	Virtual Device Metafile

APPENDIX A

Statement of Work TCN:84-108

STATEMENT OF WORK
TCN: 84-108

US ARMY ENGINEER TOPOGRAPHIC LABORATORIES
FORT BELVOIR, VIRGINIA 22060

ETL-TD-MA

26 Jan 84

Scientific Services Program - STAS

1. TITLE: Study of Raster Metafile Formats

2. GENERAL

The services of a scientist are required to study metafile formats suitable for use with raster graphics systems. This study is required so that a prototype metafile format can be identified which may be incorporated in a raster device-independent graphics software system. The necessary in-house capability is not available to perform this task.

A metafile is a standard device-independent display-record format. The purpose of a metafile is to:

- o provide a universal method for transferring graphic images between two computing sites
- o provide an audit trail of image development
- o provide a data source for hardcopies of images designed during interactive graphics sessions
- o provide an archival medium
- o assist in certification and verification of graphics data
- o serve as interface standard for intelligent peripherals.

In addition, a metafile designed for use with raster graphics systems must be capable of handling scanned images, text, and synthetically generated graphics.

3. OBJECTIVE

The objective of this short term analysis is to study metafile formats suitable for use with raster graphics systems. A report examining raster metafile formats will be delivered at the conclusion of the study.

4. SPECIFIC TASKS

The principal issue associated with raster metafile formats is developing a format which can handle each of the common raster data types associated with various raster device types. The data types include scanned images, text, and synthetically generated graphics. The raster device types include full color raster display systems, high resolution black-and-white printers, and raster printers.

Attachment 2

a. The scientist shall perform a qualitative analysis of proposed and adopted standard device-independent display-record formats to identify potential viable raster metafile formats.

b. The scientist shall review the Defense Mapping Agency report on raster reformatting and describe the applicability of the report for use in the definition of raster metafile formats.

c. The scientist shall develop a proposal for a standard raster metafile format.

5. REPORTING REQUIREMENTS

a. Oral reports as required.

b. Brief written progress reports shall be mailed to the COTR on the first of each month during the period of performance.

c. A final typewritten report (2 copies) summarizing the work and conclusions derived shall be submitted to the COTR within 30 days after completion of services. The report shall also include a bibliography of past work in the area.

6. QUALIFICATION REQUIREMENTS

The principal scientist or engineer selected for these services must have the educational equivalent of a Ph.D. in computer science or electrical engineering. The principal individual must have specific experience in computer graphics, raster-based computer graphics, and the definition of raster graphics standards. Assistant scientists must have specific experience in computer graphics, raster-based computer graphics, and the definition of raster graphics standards.

7. PLACE AND PERIOD OF PERFORMANCE

a. A total of 48.5 working days for a research group are required during the 6 month period beginning with EDODO.

b. Two trips to the US Army Engineer Topographic Laboratories are planned.

c. Approximately 95 percent of the work should be performed at the scientist's facility, the remaining time to be spent at Government facilities.

8. RESTRICTIONS

There is no known potential conflict of interest associated with this task.

9. SECURITY CLEARANCE

No clearance is required.

10. COTR

Richard L. Rosenthal
US Army Engineer Topographic Laboratories (ETL-TDL)
Fort Belvoir, VA 22060
Autovon 354-3722
Commercial (703) 664-3722

APPENDIX B

Summaries of 840913 and 841120 project meetings

Summary of 840913 meeting at the
Engineer Topographic Laboratories (ETL)
Fort Belvoir, Virginia
(prepared by Marshall Taylor)

On Thursday, September 13, 1984, Marshall Taylor and Peter French of RPA meet for two and a half hours with ETL and Defense Mapping Agency Hydrographic / Topographic Center (DMAHTC) staff. Discussions were held on the progress of the study of raster metafiles contracted by RPA under the Scientific Services Program. Present, in addition to RPA staff were:

Mr. Richard Rosenthal - ETL staff member previously assigned as COTR for the project,
Ms. Rose Holcheck - ETL staff member serving as COTR, and,
Mr. Art Noma - DMAHTC staff member.

I gave a brief presentation of our progress to date and then a brief overview of proposed graphic standards concentrating on the proposed ANSI-VDM. Included in the discussion was a report on the phone conversations with Dr. Peter Bono, chairman of the ANSI X3H3 committee, and with Mr. Jim Kearney, the U.S. Army representative to the committee. I described the proposed "registered extension" procedures and lead a discussion on the characteristics of VDM and the possibility of ETL/DMAHTC developing significant, raster oriented extensions to that system.

Art Noma and Richard Rosenthal then led a discussion of the needs at ETL and DMAHTC. A major concern was the transporting of application programs to new hardware configurations, much as presented in the RRS report by Synectics Corporation. DMAHTC currently has several types of hardware and anticipates additional purchases with corresponding system changes.

It was pointed out that one of DMAHTC's products is three dimensional surfaces represented as cell arrays (DEM's). Any proposed raster metafile system must provide for the transfer of this type of information.

At DMA they currently are developing a test system which uses hardware-specific calls for their applications programs. It was suggested that a conversion should be made to GKS (or some other "standard") at least upon completion of the test system. Rich mentioned that some existing hardware currently had "nice" features included as firmware, and that such surely will be the case for future graphics hardware. It was recognized that if a standard such as GKS were used, some of these special features may not be accessible, at least through GKS. (Note: ETL might want to use a standardized non-standard call to invoke those special functions or capabilities currently

being used. This would cause those functions currently running in firmware to run more slowly, but the flexibility provided in terms of application software transportability would make it worthwhile. Also, consider the possibility of making the call generic enough so that it might be submitted as a possible registered extension to ANSI - GKS or VDM.)

Rich mentioned the need to consider input as well as output. Currently, raster plotters can be used in single bit mode to produce drawings at 60000 by 40000 pixels, vs. "soft" displays on monitors (1024 x 1024 x 24 bits). Rich suggested that we "shoot for the stars" in terms of what we think might eventually need to be done, but also discuss where the industry currently appears to be headed.

There was a considerable amount of discussion about the differences between geographic-cartographic-graphic information/displays. The major conclusion was that ETL/DMAHTC have, or are developing, advanced cartographic data standards and cartographic data base management systems. There is no need for a proposed graphic metafile system to in any way duplicate this effort.

The question was raised as to whether or not additional information (such as cell arrays or additional geographic-related information, as headers, should be included in a proposed metafile system). Specifically, ETL/DMAHTC were asked if they viewed it necessary to store results from one set of analyses so that a partial solution, or starting point, could be passed as a VDM to another device having different or additional applications software. The comments here were again related mainly to the use of cartographic data standards and the separability of the graphic metafile question.

Regarding the use of VDM, Rich agreed that the use of a standard VDM escape to permit the introduction of non-standard information would be very useful. This could include graphic information (i.e. pix file or some type of compacted data) as well as non-graphic information. He believes that including non-graphic information, such as headers containing origination, scaling, coordinates, etc., could be useful. Rich said that we should address the ability to add headers, versus specifying (as did Synectics) what those headers should contain.

Art stated that he essentially ran a map factory, and that we should address the main production line where their quota is so many maps/time period. He viewed the use of a VDM type of system as necessary for special products. These may be unique, one-time images with associated headers, etc. Art expressed a need for a critique of the Synectics report with regard to a metafile in our completion report.

Rich said that DMA uses a standard linear format, so that a graphics metafile is only needed at the input end or at the output (product) end. Metafiles would not be used to feed interim processes. (Note: there may be a distinction between feeding the interim processes and insuring the ability to represent graphic data on any graphic device regardless of the device on which it was originally created.)

A discussion between Art and Rich then ensued as to the need to be able to mix text (including symbols) with the graphics, but as separate entities. Rich indicated that if the symbols were present as part of the graphic (i.e. visible when the cell array data were displayed), that would be sufficient. Art said that different character symbols (A-Z, 0-9, ASCII chars., plus foreign language characters, e.g. Arabic, Farsi, etc.) were permitted in terms of storing "text", so why not be able to store special symbols (e.g. swamps). Rich indicated that this application may fit within the definition of "markers" used for GKS and VDM.

A final discussion involved the method/ability to represent polygonal information, especially if the data contained holes (or islands within islands).

Additional notes:

1) DMAHTC's charter explicitly prevents it from conducting research. It uses ETL, Rome ADC, and _____ as research arms.

2) We should follow-up with Dr. Bono regarding "registered" extensions.

3) Clearly differentiate the "object definition" metafile approaches (IGES) with the graphic metafile approaches such as VDM.

Summary of 841120 meeting at the
Engineering Topographic Laboratories (ETL)
Fort Belvoir, Virginia

(prepared by Marshall Taylor)

On Tuesday, November 20, 1984, I met for over two hours with ETL and Defense Mapping Agency Hydrographic/Topographic Center (DMAHTC) staff. Discussions were held on the draft technical report for the study of raster metafiles. Present, in addition to myself, were:

Mr. Larry Cook - ETL staff member and current COTR,
Mr. Art Noma - DMAHTC
Mr. Richard Rosenthal - ETL
Mr. Dave Scott - ETL

- I began the discussion with a brief, point by point, summary of the conclusions and recommendations contained in our draft technical report. Each of the items listed on the conclusions and recommendations page of that draft report were covered. Particular attention was focused on the first four items.

- Ensuing discussion about the role of a raster metafile indicated that our final report must clearly identify the portion of DMAHTC's overall concerns with which the report deals. In addition, the final report should have an expanded discussion of the distinctions between "product definition level" metafiles and graphic metafiles such as VDM. Examples of the application of each level of metafile would improve the report. For clarity we should insure that the conclusions refer to the problem identification earlier described.

- Both Art and Rich suggested that we choose one functional area in which to give an example of embedding RRS capabilities into VDM. Perhaps the best way to do this is to expand the technical report's section on pixel encoding. This could also serve to begin putting some hard numbers to the data and computation cost of adopting the more flexible VDM specifications over the more efficient RRS formats.

- Art Noma mentioned that current specifications for a raster scanner being acquired by DMA include a requirement for RRS compatibility. I stated that DMAHTC should not proceed with placing RRS on each of their devices. Art indicated that he agreed.

- Questions were raised as to how ETL/DMAHTC could effectively participate in ANSI X3H3 and related committees. I suggested

that the question be addressed directly to Dr. Bono. Before the final presentation of our report we should insure that ETL has followed up on this or contact Peter Bono ourselves. In addition, we may want to review any informal ties between the National Committee for Digital Cartographic Data Standards and ANSI X3H3 or other committees.

- Art and Rich suggested that we write up brief descriptions of further studies which we believe they should undertake. These would include:

    a. Examination of IGES level metafiles and the relations between various metafile levels,

    b. Examination of GKS/PHIGS as a programming standard for ETL/DMAHTC and identification of benefits and possible problems, and

    c. Development of implementation guideline and specifications for adopting VDM as the ETL/DMAHTC graphic metafile standard.

Art Noma wants a presentation of the final report given at DMAHTC. Discussions indicated that this presentation should be made in January. Since our project has a deadline in late December, Larry Cook agreed to ask for an extension until 1/31/85. In the mean time Larry agreed to have the draft report throughly reviewed and comments sent up to us within a couple of weeks. I will have to contact Larry concerning scheduling the January visit and finish up the final report right after the first of the year.

APPENDIX C

Draft Proposed American National Standard
Virtual Device Metafile, X3.122-198x

X3.122-198x

Project: 347

Draft Proposed

American National Standard

Virtual Device Metafile ←

This draft standard is published for a four-month period of public review and comment and subsequent letter ballot of American National Standards Committee X3. Comments received during this period will be considered and answered. Commentors who object to approval of this draft standard as an American National Standard should so indicate including their reasons.

All comments should be returned as soon as possible but not later than May 6, 1984 to:

X3 Secretariat/CBEMA
311 First Street, N. W.
Suite 500
Washington DC 20001

A copy of the comments should be sent to:

Board of Standards Review
American National Standards Institute
1430 Broadway
New York NY 10018

Prepared by

Technical Committee X3H3 - Computer Graphics

American National Standards Committee

X3 - Information Processing Systems

Secretariat: Computer and Business Equipment Manufacturers Association

dpANS
Information Processing
Computer Graphics
Virtual Device Metafile (VDM)
Functional Description

American National Standard
Virtual Device Metafile

ABSTRACT

The Virtual Device Metafile (VDM) is a set of basic elements for a computer graphics data interface usuable by many graphics-producing systems and applications. This Standard:

- allows graphics data to be easily transported between computer graphics devices and installations.
- aids computer graphics software implementors in understanding and using graphics data storage methods.
- guides device manufacturers on useful graphics capabilities.

This standard standardizes a presentation level interface to a graphics system. Hence, it contains elements for:

- graphical output primitives
- controlling the appearance of graphical primitives with attributes.
- metafile interpretation.
- setting the interpretation modes of attribute elements.

FORWARD

(This foreward is not part of American National Standard Computer Graphics Virtual Device Metafile, X3.nnn-198n)

This American National Standard provides a set of basic elements for computer graphics data. These functions taken as a whole are called the Virtual Device Metafile (VDM). The design of this standard is based on the work of many groups. Much of the early design methodology was heavily influenced by the work of the Graphics Standards Planning Committee of the Special Interest Group on Computer Graphics of the Association for Computing Machinery (ACM-SIGGRAPH GPSC). This work, known as the CORE SYSTEM, contained a metafile proposal and was widely distributed in 1979. The VDM itself was originally developed by X3H3 in 1981 and was subsequently refined extensively during the period 1982-1983 in cooperation with Working Group 2 of the Subcommittee on Programming Languages of the Technical Committee on Information Processing of the International Standards Organization (ISO TC97/SC5/WG2).

This standard was developed by Technical Committee X3H3 of American National Standards committee X3 under project 347I, started in March, 1981. The VDM was approved as an ISO work item (97.5.19) in May, 1983. This draft, which is currently out for public review within ANSI, will be registered as a DPIS and subjected to ISO review. Changes resulting from either the ISO review or the ANSI public review will be reflected in both documents. Certain elements of the VDM are included to provide support for the Graphical Kernal System (GKS). X3H3 will track the changes that occur in GKS (ISO DIS 7942) and integrate those changes that are appropriate.

This standard was approved as an American National Standard by the American National Standards Institute on mmmm, dd, 198y.

Suggestions for improvement of this standard will be welcome. They should be sent to the American National Standards Institute, 1430 Broadway, New York, N.Y. 10018.

This standard was processed and approved for submittal to ANSI by American National Standards Committee on Computers and Information Processing, X3. Committee approval of the standard does not necessarily imply that all committee members voted for its approval. At the time it approved this standard, the X3 Committee had the following members:

XXXXXXXXXX, Chairman
XXXXXXXXXX, Vice-Chairman
XXXXXXXXXX, Secretary

Organization Represented	Name of Representative
ABC, Inc.	J. M. Doe

Technical Committee X3H3 on Computer Graphics, which developed the draft proposals, which held the U.S. Technical Advisory Group responsibilities for ISO TC97/SC5/WG2, and through which this standard was completed, had the following members:

P. Bono, Chairman
B. Shepherd, Vice-Chairman
B. Simons, Secretary
J. Chin, International Representative

Organization Represented	Name of Representative(s)
Amoco Production	M. Whyles
Athena Systems	P. Bono
Benson	S. Gill
	D. Slaby (alt)
Bell Northern Research	C. Mannhardt
Calma	W. Dale
Computer Sciences	D. Lynch
	J. Reese (alt)
	K. Leung (alt)
Control Data Corporation	C. Seum
	J. Schoenburg (alt)
Data General	J. Hargrove
	K. Kimbrough (alt)
	D. Galewsky (alt)
Dept. of Communications, Canada	N. Newman
Digital Equipment Corp.	T. Powers
	J. McConnell (alt)
	R. McNall (alt)
Digital Research	F. Langhorst
	R. Kan (alt)
Evans & Sutherland	R. Ehlers
Fortune Systems	J. Chin
Graphic Software Systems	R. Perry
	E. Post (alt)
	T. Clarkson (alt)
	G. Cuthbert
Harris	T. Morrissey
Hewlett-Packard	P. Showman (alt)
	E. McGinnis (alt)
	A. Frankel (alt)
Houston Instruments	B. Plunkett
Hughes Aircraft	R. Harney
Intel	M. Brown
	B. Olenchuk (alt)
	C. Duffy (alt)
IBM	B. Cohen (alt)
Imlac Corporation	B. Shepherd
	P. Norman
ISSCO	A. Herrick (alt)
	T. Wright
	D. Kusumoto (alt)

Jet Propulsion Laboratory	L. Pfeheim
	R. Holzman (alt)
Laurence Berkeley Laboratory	W. Johnston
	D. Cahn (alt)
Laurence Livermore Laboratory	J. Rowe
Los Alamos National Laboratory	T. Reed
	R. Elliott (alt)
McDonnell Douglas Automation Co.	D. Shuey
Megatek	M. Mock
	M. Plaehn (alt)
	M. Languth (alt)
	R. Bruns (alt)
Microsoft	J. Butler
	D. McCabe (alt)
Mindset	J. Blair
NCAR	L. Henderson
NBS	M. Skall
Norpak	O. Lapczak
NSA	K. Schmucker
Olivetti	P. Jones
	A. Leinwand (alt)
Precision Visuals	M. Babcock (alt)
	M. Journey
	S. Stash (alt)
Puk Consulting	R. Puk
RPI	A. Bunshaft
SAI	R. Stout
	T. Thornton (alt)
Sanders Associates-Calcomp	D. Bailey
	K. Hepworth (alt)
Sandia	R. Simons
SDC	M. Sparks
	L. Benbrooks (alt)
SIGGRAPH	E. Sonderegger
	L. Hatfield (alt)
Sperry Corporation	N. Soong
	T. Mainock (alt)
	W. Hafner (alt)
Systonetics, Inc.	F. Canfield
Tektronix	D. Strasyer
	B. Ross (alt)
	G. Strockbine
Tymshare, Inc.	J. Kearney
U.S. Army	R. Flippen (alt)
	S. Carson
WPL, Inc.	J. Bedrick
Wang Lab	W. Yip (alt)
	B. Sangster (alt)

CONTENTS

TABLES

FIGURES

Part 2. VIRTUAL DEVICE METAFILE CHARACTER ENCODING

TABLES

FIGURES

Part 3. VIRTUAL DEVICE METAFILE BINARY ENCODING

TABLES

Part 1. VIRTUAL DEVICE METAFILE FUNCTIONAL DESCRIPTION

1. SCOPE, FIELD OF APPLICATION, AND BENEFITS

1.1 Scope

This standard establishes the methodology, functional elements, and encoding of the Virtual Device Metafile (VDM). This standard defines a single, usable set of VDM elements that is expected to satisfy the following needs of a majority of the computer graphics community:

(1) Provide a graphics data interface standard for computer graphics software package implementors.

(2) Provide a picture transfer standard between computer graphics devices.

(3) Provide a picture transfer standard between computer graphics installations.

(4) Provide a picture transfer standard between computer graphics systems.

(5) Provide a standard graphics escape mechanism to access nonstandard graphics device capabilities.

(6) Provide for future functional extension of the VDM.

This document is limited to the definition of the Virtual Device Metafile (ANSC X3 project number 347D). The VDM is intended to be a graphics data interface. The VDM is closely related in functional capability to the proposed Virtual Device Interface standard (ANSC X3 project number 346D).

The minimal capability of this VDM includes all elements necessary to describe pictures independently of each other. The pictures in the VDM are storable on different media, transportable between different graphics systems, and displayable on different graphics devices. After processing the METAFILE DESCRIPTOR, pictures in the VDM are correctly interpreted with either sequential processing from the beginning of the VDM, or with random access to the picture boundary and sequential access to elements within the definition of the picture.

The specific mechanisms of metafile generation and interpretation are not described by this standard, although it does describe the intended result of such interpretation. The basic set of VDM elements includes a capability for the addition of application-dependent data.

VDM functionality is separated from the specification of any particular encoding format. Two generally useful encoding formats are included as part of this standard. Others may be developed in the future.

1.2 Field of Application

The project proposal for the Virtual Device Metafile defined the VDM and its field of application. The following statements from the proposal are repeated here:

(1) "The VDM is a mechanism for retaining and/or transporting graphics data and control information."

Retention and transportation may imply the *separation*, in time or space, or both, of the process that creates the VDM and the process using it. This separation of processes and its implications for assumptions by one process about the characteristics of the other have been considered carefully while producing this metafile standard.

(2) "This information contains a device-independent description of a picture..."

Device independence inherently limits the set of elements that can be included in a metafile. A standard mechanism (VDM ESCAPE) is provided to permit access to device-dependent features.

(3) "The VDM is at the level of the Virtual Device Interface."

The VDM elements are defined consistently with those of the Virtual Device Interface. This effort defines a minimal, useful set of elements sufficient to describe device-independent graphics pictures. There are several reasons supporting development of this metafile with minimal, but sufficient, capability. For example,

- It encourages early acceptance and implementation.
- It is easier to implement.
- Its concepts are compatible with existing practice.
- It provides a "low-overhead mode of operation".

Advanced concepts such as segmentation, macro facilities, three-dimensional elements, and others are not included but are also not precluded from future extensions to this standard.

1.3 Benefits

1.3.1 Intrinsic. This VDM allows portability of graphics data among installations. This standard encourages a uniform interface for noninteractive picture description.

1.3.2 Interchangeable. This standard promotes the exchange of information that enables installations to share work and reduce time spent recomputing in an effort to regenerate graphics data. The standard enables media (for example, magnetic tape) or communication (for example, line protocol) transfer of data.

1.3.3 Educational. This standard set of elements uses standard terminology, which allows both the academic and industrial communities to develop instructional programs that concentrate on programming techniques and methodologies based on these standard elements.

1.3.4 Economic. The major economic benefit is derived from defining a unified external format for graphics data, thus allowing the same graphics data to be displayed on different devices. The following benefits will be derived from this standard:

- Benchmarks can be run on different vendors' equipment.
- Graphics output can be recorded as an aid in debugging.
- Archiving, off-line plotting, and off-site plotting can take place.
- Animation sequences can be built in nonreal time and nonsequential order, and then viewed in real time in the proper sequence.
- Selected pictures can be previewed before a large number of pictures are sent to a more expensive or slower medium.
- A standard interface can be developed for a variety of plotting, COM, and other off-line, picture-generating devices.
- The same picture or series of pictures can be used several times without recalculating the picture.

2. REFERENCES AND RELATED EXISTING STANDARDS

2.1 Existing ANSI Standards

This metafile standard is a graphical picture file exchange standard and not a product definition database exchange standard. Standards work in the latter area is the responsibility of the ANSC Y14.26 (Computer-Aided Preparation of Product Definition Data) Technical Committee. The VDM standard is concerned with the generation and transfer of sufficient device-independent information for a picture to be drawn on a wide variety of graphics output devices. The ANSI Y14.26M-1981 standard ("IGES") is concerned with the transfer of all product definition data (geometric and nongeometric) across CAD/CAM systems. Specifically, ANSI Y14.26M-1981 and this standard deal with different information for different purposes at different levels of detail.

2.2 ANSI Standards Development Projects

The Virtual Device Metafile (VDM) standard has been developed in conjunction with the higher level standards being developed by ANSC X3H3 (Computer Graphics Programming Language). Coordination within X3H3 has taken place in order to achieve consistency among the interrelated standards. The development of the Virtual Device Interface (VDI) by X3H3 has been closely related to the development of this metafile standard. Also, coordination with work of the ANSC X3L2 (Character Sets and Coding), ANSC X3J6 (Text Processing Language), and ANSC X3J7 (APT) Technical Committees has taken place. While there are similarities between the VDM and the North American Presentation-Level Protocol Syntax (NAPLPS: ANSI X3.110-1983), the latter is designed to support a particular class of devices in a picture transmission environment, while the VDM is intended to provide picture definition in a device-independent and environment-independent manner.

2.3 ISO Standards Projects

This standard has been developed in collaboration with ISO TC97/SC5/WG2 under project 97.5.19 authorized in May 1983. The Graphical Kernel System (DIS 7942) specifically excluded portions pertaining to metafiles in anticipation of this metafile standard. ISO TC97/SC18 (Text Preparation and Interchange) is developing a standard on text imaging capabilities which includes the specification of graphical elements and attributes. Its work has been considered where applicable. Coordination with the work of ISO TC97/SC2/WG8 (Picture Coding) has taken place. It is expected that one use of the VDM will be as a GKS Metafile at level 0a of GKS.

2.4. IEEE Standards Projects

A Proposed Standard for Binary Floating-Point Arithmetic (Draft 10.0 of IEEE Task P754) is used for the floating-point representation within the binary encoding.

3. TERMINOLOGY

3.1 Definitions

ASPECT RATIO
The ratio of the width to the height of a rectangular area, such as a window or viewport. For example, an aspect ratio of 2.0 indicates an area twice as wide as it is high.

ASPECT SOURCE FLAG (ASF)
Indicator (Flag) as to whether a particular attribute selection is to be individual or bundled.

ATTRIBUTE ELEMENTS
Metafile elements that describe the appearance of graphical elements.

BUNDLE
Set of attributes associated with one of the following graphical element types: POLYLINE, POLYMARKER, TEXT, and Filled Area.

BUNDLE INDEX
Index for accessing a particular set of attributes in a bundle table.

BUNDLE TABLE
An indexed table containing a set of attributes for each index.

CLIP INDICATOR
Indicator (Flag) as to whether metafile graphical elements are to be clipped at the limits of CLIP RECTANGLE.

CLIP RECTANGLE
A rectangle defined in VDC space which is used as a clipping boundary when the metafile graphical elements are to be clipped.

CLIPPING
The process of removing any portion of a graphical image which extends beyond a specified boundary.

COLOR SELECTION MODE
Indicator (flag) as to whether color selection is to be direct (by specifying the RGB values) or indexed (by specifying an index into a table of RGB values).

COLOR TABLE
A table for use in mapping from a color index to the corresponding color. See DIRECT COLOR, INDEXED COLOR.

COLOR VALUE
The values of the RGB (red, green, blue) components describing a color.

CONTROL ELEMENTS
Metafile elements that specify metafile delimiters, address space, clipping boundaries, picture delimiters, and format descriptions of the VDM elements.

DATA INTERFACE
An interface between software modules or devices comprising one or more packets containing opcodes and data--as contrasted with a subroutine call interface.

DESCRIPTOR ELEMENTS
Metafile elements that describe the functional content, format, default conditions, identification, and characteristics of the VDM.

DEVICE DRIVER
The device-dependent part of a graphics implementation which supports a physical device. The device driver generates device-dependent output.

DIRECT COLOR
A color selection scheme in which the color values are specified directly, without requiring an intermediate mapping via a color table. See COLOR TABLE, INDEXED COLOR.

DISPLAY SURFACE, VIEW SURFACE
That part of a graphics device upon which a visible image appears (for example, the screen of a display, the paper in a plotter).

DOT
The smallest visible point that can be displayed on the display surface. See PIXEL.

ESCAPE ELEMENTS
Metafile elements that describe device- or system-dependent elements used to construct a picture, but that are not otherwise standardized.

EXTERNAL ELEMENTS
Metafile elements that communicate information not directly related to the generation of a graphical image.

GRAPHICAL ELEMENTS
Metafile elements that describe images in the VDM.

GRAPHICAL KERNEL SYSTEM (GKS)
A standardized application programmer's interface to graphics. Refer to ISO/DIS 7942 and ANSI dpANS X3H3/83-25R2.

GRAPHICS DEVICE
A device (for example, refresh display, storage tube display, or plotter) on which display images can be represented.

HATCH STYLE
A format for filling closed figures. A hatch style consists of one or more sets of lines whose presence represents the interior of the figure in question.

INDEXED COLOR
A color selection scheme in which the color index is used to retrieve color values from a color table. See COLOR TABLE, DIRECT COLOR.

MESSAGE
A string of characters used to communicate information to operators at Metafile interpretation time.

METAFILE
A mechanism for retaining and transporting graphical data and control information. This information contains a device-independent description of one or more pictures.

METAFILE ELEMENT
A functional item that can be used to construct a picture or convey information.

NORMALIZED DEVICE COORDINATES (NDC)
Coordinates specified in a device-independent coordinate system, normalized to some range (typically 0 to 1). See VDC EXTENT, VDC RANGE, VDC SPACE, VIRTUAL DEVICE COORDINATES.

PATTERN STYLE
A format for filling closed figures with patterns. A pattern style consists of an array of variously colored or shaded points.

PICTURE DESCRIPTOR ELEMENTS
Metafile elements used to set the interpretation modes of attribute elements for the entire picture.

PIXEL
The smallest element of a display surface which can be independently assigned color. See DOT.

VIRTUAL DEVICE
An idealized graphics device that presents a set of graphics capabilities to graphics software or systems via the Virtual Device Interface.

VIRTUAL DEVICE COORDINATES (VDC)
The coordinates used to specify position in the VDC space. These are absolute two-dimensional coordinates. See VDC SPACE.

VDC EXTENT
A rectangular region of interest contained within the VDC range. See VDC RANGE, VDC SPACE.

VDC RANGE
A rectangular region within VDC space consisting of the set of all coordinates representable in the declared coordinate type, precision, and encoding format of the metafile. See VDC EXTENT, VDC SPACE.

VDC SPACE
A two-dimensional, Cartesian coordinate space of infinite precision and extent. Only a subset of VDC space, the VDC range, is realizable in a metafile. See VDC EXTENT, VDC RANGE, VIRTUAL DEVICE COORDINATES.

VIRTUAL DEVICE INTERFACE (VDI)
An interface between the device-independent and the device-dependent levels of a graphics system. This interface may be implemented in either a device driver or a device.

VIRTUAL DEVICE METAFILE (VDM)
A mechanism for retaining and transporting graphics data and control information at the level of the Virtual Device Interface.

VDM GENERATOR
The process or equipment that produces the metafile.

VDM INTERPRETER
The process or equipment that reads the metafile and interprets the contents. An interpreter may be needed in order to drive a Virtual Device Interface or other device interface to obtain a picture that resembles the intended picture as closely as possible.

3.2 Abbreviations:

ANSC	American National Standards Committee
ANSI	American National Standards Institute
ASF	Aspect Source Flag
GKS	Graphical Kernel System
NDC	Normalized Device Coordinates
VDC	Virtual Device Coordinate
VDI	Virtual Device Interface
VDM	Virtual Device Metafile

4. VIRTUAL DEVICE METAFILE CONCEPTS

4.1 Introduction

The objectives of the Virtual Device Metafile (VDM) are to provide for the description, storage, and communication of graphical information in a device-independent manner. To accomplish these objectives, this standard defines the form (syntax) and functional behavior (semantics) of a set of elements that may occur in the VDM. There are six classes of elements.

- Descriptor Elements, which describe the functional content, default conditions, identification, and characteristics of the VDM.

- Control Elements, which specify metafile delimiters, address space, picture delimiters, and format descriptions of the VDM elements.

- Picture Descriptor Elements, which set the interpretation modes of attribute elements for the entire picture.

- Graphical Elements, which describe the visual components of a picture in the VDM.

- Attribute Elements, which describe the appearance of graphical elements.

- Escape Elements, which describe device- or system-dependent elements used to construct a picture; however, the elements are not otherwise standardized.

- External Elements, which communicate information not directly related to the generation of a graphical image.

A Virtual Device Metafile is a collection of elements from this standardized set. The full set or proper subsets of the standardized elements may be represented in the collection. Such a metafile must be interpreted in order to display its pictorial content on a graphics device. The Descriptor Elements give the interpreter sufficient data to interpret metafile elements and to make informed decisions concerning the resources needed for display.

The elements in a metafile may include control elements for metafile interpretation, picture descriptor elements for declaring parameter modes of attribute elements, graphical elements for defining graphical entities, attribute elements for defining the appearance of the graphical elements, escape elements for accessing nonstandardized features of particular devices, and external elements for communication of information external to the definition of the picture in the VDM.

Initially two bindings or encodings are specified by this standard: a character-coded binding and a binary binding. Other bindings may be developed, either for private use or as an extension to this standard. All such bindings must conform to the functionality specified in this document.

4.2 Descriptor Elements

The Metafile Descriptor (MD) is a group of elements that describes the functional capabilities required to interpret the VDM. In a particular metafile, the VDM ELEMENTS LIST lists at least those standardized elements that occur in the metafile. The VDM interpreter is thus informed of the capabilities required to successfully interpret the Virtual Device Metafile. The VDM must contain a single Metafile Descriptor. The Metafile Descriptor must be located immediately after the BEGIN METAFILE element in a metafile (with the possible exception of intervening external and escape elements).

4.2.1 Identification. The identifying information includes declaration of the version of the VDM standard and descriptive information about the origin, owner, generation date, etc., of the metafile.

4.2.2 Characteristics. The characteristics provide information such as the functional capabilities required to interpret the metafile and the changes to the default state of the VDM interpreter. The contents of the Virtual Device Metafile are defined by a list of the control elements, graphical elements, and attribute elements that are utilized in the metafile. The default state is the state to which the interpreter is returned at the start of each picture. The default states of all metafile elements are defined in Section 6. These default values may be selectively replaced by using the VDM DEFAULTS REPLACEMENT element. The correspondence between character set indexes and registered or private character sets, and the meaning assigned to text font indexes, are also established in the Metafile Descriptor.

4.3 Control Elements

Control elements specify metafile delimiters, address space, clipping boundaries, picture delimiters, and format descriptions of the VDM elements. Control of some of these format descriptions may be accomplished by Metafile Descriptor elements, while control of others is accomplished by control elements in the body of the metafile. Those items in the former category are fixed for a given metafile, while those in the latter category are changeable; that is, they may change in the body of a picture.

4.3.1 VDC Space and Range. The graphical elements of a metafile define virtual images. The coordinates of these elements (that is, the addresses of points in the virtual image) are absolute two-dimensional Virtual Device Coordinates (VDC). VDC space is a two-dimensional Cartesian coordinate space of infinite precision and infinite extent. Only a subset of VDC space, the VDC range, is realizable. The VDC range comprises all coordinates representable in the format specified by the declared VDC TYPE and VDC PRECISION found in the metafile.

The VDC range is not directly settable; it is completely determined by VDC TYPE and either INTEGER VDC PRECISION or REAL VDC PRECISION elements in the metafile. These elements are controllable, some by dynamic elements in the metafile body and some by static elements in the MD. Note that the VDC range thus defined (a rectangular subregion of the VDC space) does not enclose a continuum of values, but has a distinct granularity. Regardless of the aspect ratio of the VDC range and the granularity within the range, it is implicit that one address unit in the x-direction represents the same distance as one address unit in the y-direction in VDC space.

4.3.2 VDC Extent. There is a metafile element to define the VDC extent. The extent is set by specifying the addresses (in VDC) of the lower-left corner and the upper-right corner of this extent as seen by the viewer of the picture. Specification of values outside VDC EXTENT is permitted in VDM elements. It is intended that the visible portion of an image be contained within VDC EXTENT. It thus provides a frame for the region of interest in a picture. The values of the coordinates for either dimension may be either increasing or decreasing from the first to the second corner. For example, for devices with an upper-left origin, a picture may be described in coordinates that map directly to the device but still may be displayed correctly on a device with a lower-left origin. Figure 4-1 illustrates these concepts.

The VDC extent thus establishes the sense and orientation of VDC space (that is, the directions of the positive x (+x) and positive y (+y) axes, and whether the +y axis is 90-degrees clockwise or 90-degrees counterclockwise from the +x axis). In particular, VDC EXTENT establishes the direction of positive and negative angles as follows: positive 90-degrees is defined to be the right angle from the positive x-axis to the positive y-axis (see Figure 4-1). Note that some attributes such as text attributes (for example, the directions of the 'up' and 'base' component vectors of CHARACTER ORIENTATION, and therefore the meaning of the enumerative values 'right', 'left', 'up', 'down') are intimately bound to these definitions.

The default state of the extent is specified in Section 6 and can be changed in the VDM DEFAULTS REPLACEMENT element in the MD [illegible] to this default state at the beginning of each picture.

4.3.3 Clipping. In order to defer clipping of graphical elements (particularly expandable elements such as CIRCLE, ARC, TEXT, etc.) until metafile interpretation time, a clipping control feature is provided in the VDM. Clipping control is achieved by defining CLIP RECTANGLE in VDC space. Whether clipping to the limits of CLIP RECTANGLE actually occurs at metafile interpretation time is controlled by the CLIP INDICATOR element that sets the mode of the metafile to 'clipping on' or 'clipping off'. The defaults for CLIP RECTANGLE and CLIP INDICATOR are listed in Section 6.

4.3.4 VDC Tailoring. The ability to specify the VDC range and the VDC extent provides the flexibility to configure the metafile effectively in any way desired. It can be configured as an abstract, normalized address range for maximum device independence. It can also be configured to mimic the addressability of a particular target device in order to take advantage of particular device characteristics. The address range of such a device-specific metafile is just another normalized address range with the normalization limits inherent in the VDC scaling element. Therefore, device independence is maintained.

Such tailoring of the coordinates in a metafile can eliminate the need for transformation of coordinates at metafile interpretation time for the target device. The ability to specify VDC extent allows for the exact registration of coordinates in a metafile with addressable points on the target graphics device. Also, note that a metafile generator can limit or tailor the functional content of a metafile to accommodate particular devices or applications through the use of VDM ELEMENT LIST.

4.4 Picture Descriptor Elements

Picture descriptor elements are those elements that declare the parameters for all other elements for an entire picture. These elements are SCALING MODE, COLOR SELECTION MODE, LINE WIDTH SPECIFICATION MODE, MARKER SIZE SPECIFICATION MODE, and PERIMETER WIDTH SPECIFICATION MODE. If included in a picture, they must appear after the BEGIN PICTURE element and before the BEGIN PICTURE BODY element. Escape and external elements are permitted in the picture descriptor.

4.4.1 Scaling Mode. VDC space may be either an abstract space, which may be mapped to an arbitrary size on a physical device, or a metric space, which is intended to be mapped to a particular size. Selection of the mode to be used can be made on a picture-by-picture basis by means of the SCALING MODE element. The scaling mode element provides a flag to select abstract space or metric space, and a scale factor which specifies the number of millimeters per VDC unit when metric space is selected.

4.4.2 Color Selection Mode. COLOR SELECTION MODE selects either indexed or direct (RGB) color specification for the picture and is described further under color attributes.

4.4.3 Specification Modes. Line width, marker size, and perimeter width may be specified in more than one way. The width of lines, for example, may be specified as either a measure in VDC units or as a scale factor to be applied to a device-dependent nominal line width at interpretation time. For each attribute element having such multiple modes, there is an associated control element that defines the mode of the parameter of the attribute element.

4.5 Graphical Elements

Graphical elements are those elements that describe the visual components of a picture. They are specified in VDC units. The VDM provides the graphical elements POLYLINE, POLYMARKER, POLYGON, CIRCLE, ARC, ARC CLOSE, TEXT, APPEND TEXT, and CELL ARRAY.

The formal specification of the VDM does not contain the concept of current position (CP). Therefore, all graphical elements are logically independent. Because the metafile does not contain the concept of CP, metafile interpreters may drive graphics devices without consideration of the relationship of a given CP model to a given device. However, encoding of this metafile standard can be produced to realize the efficiency advantages often associated with CP.

The TEXT and APPEND TEXT elements and related text attribute elements are defined in the current VDC space. Thus, they are affected by changes to the Virtual Device Coordinate format.

4.6 Attribute Elements

Attribute elements describe the appearance of graphical elements. Attributes are classified as either individual or bundleable. Table 4-1 lists the individual and bundleable attributes.

Table 4-1: Individual and Bundleable Attributes

Individual	Bundleable
PATTERN SIZE	LINE TYPE
PATTERN REFERENCE POINT	LINE WIDTH
CHARACTER HEIGHT	LINE COLOR
CHARACTER ORIENTATION	MARKER TYPE
CHARACTER PATH	MARKER SIZE
TEXT ALIGNMENT	MARKER COLOR
CHARACTER SET INDEX	INTERIOR STYLE
(LOCAL BACKGROUND COLOR)	FILL COLOR
	HATCH INDEX
	PATTERN INDEX
	PERIMETER TYPE
	PERIMETER WIDTH
	PERIMETER COLOR
	TEXT FONT INDEX
	TEXT PRECISION
	CHARACTER EXPANSION FACTOR
	CHARACTER SPACING
	TEXT COLOR

Note: Although LOCAL BACKGROUND COLOR is a control element, it *behaves as if it is an individual* attribute.

Bundled selection of attributes implies that the appearances of graphical elements are distinguishable from one another when different bundles are specified. The method of specification of the bundleable aspects of a primitive may be chosen *separately* for each aspect. A further group of attributes called ASPECT SOURCE FLAGS (ASFs) takes the values 'individual' and 'bundled' to specify the choice. *There is one ASF for each* bundleable aspect of each primitive.

There is a current modal value for each of the attributes. Elements are provided to change these modal values. The modal value established by setting an attribute remains until it is explicitly changed. All attributes return to their default values when the BEGIN PICTURE element is encountered.

There is a bundle specifier, the bundle index, associated with each of the graphical element types--*POLYLINE, POLYMARKER,* FILL AREA, and TEXT. The value of the bundle index for each graphical element type is modally bound to subsequent elements of that type. Distinct values of the bundle index correspond to distinct appearances of the graphical element. For each bundleable attribute, there is an associated Aspect Source Flag (ASF).

For individual attributes, the current modal value is used to display a graphical element. For bundleable attributes a graphical element is displayed as follows:

(a) If the ASF for an aspect is 'individual', the value used is the current modal value (which is set only by the individual aspect-setting elements).

(b) If the ASF for an aspect is 'bundled', the value used is obtained via the bundle table for that primitive: the corresponding component of the bundle, which is pointed to by the bundle index, is used.

The actual resulting appearance is interpreter dependent, but the intent is that the interpreter render distinct appearances for distinct values of the bundle index by manipulation of the bundleable attributes. For example, POLYLINE BUNDLE INDEX designates visually distinct combinations of the bundleable polyline attributes LINE WIDTH, LINE TYPE, and LINE COLOR. Table 4-2 lists the aspects of each bundle.

Intermixing of the individual and bundled aspects within a bundle will not allow different bundles indexes to be guaranteed to be distinguishable at interpretation time.

Table 4-2: Aspects of the Bundle and Affected Primitives

Bundle	Aspects	Affected Primitives
POLYLINE	LINE TYPE	POLYLINE
	LINE WIDTH	ARC
	LINE COLOR	
POLYMARKER	*MARKER TYPE*	POLYMARKER
	MARKER SIZE	
	MARKER COLOR	
FILL AREA	INTERIOR STYLE	POLYGON (interior and
	FILL COLOR	perimeter)
	HATCH INDEX	CIRCLE (interior and
	PATTERN INDEX	perimeter)
	PERIMETER TYPE	ARC CLOSE (interior and
	PERIMETER WIDTH	perimeter)
	PERIMETER COLOR	
TEXT	TEXT FONT INDEX	TEXT
	TEXT PRECISION	APPEND TEXT
	CHARACTER EXPANSION FACTOR	
	CHARACTER SPACING	
	TEXT COLOR	

4.6.1 POLYLINE Bundle. The POLYLINE BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) LINE TYPE: determines the type of the line (for example, 'dotted', 'dashed', etc.) with which the polyline is rendered.

b) LINE WIDTH: determines the width of the line with which the polyline is rendered.

c) LINE COLOR: determines the color in which the polyline is drawn.

4.6.2 POLYMARKER Bundle. The POLYMARKER BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) MARKER TYPE: determines the symbol that is drawn at the marker position (for example, 'dot', 'plus', etc.).

b) MARKER SIZE: determines the size of the marker symbol.

c) MARKER COLOR: determines the color in which the marker symbol is drawn.

4.6.3 FILL AREA Bundle. The FILL AREA BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) INTERIOR STYLE: determines which of the classes of interior ('hollow', 'solid', 'hatch', or 'pattern') is used to draw a filled element and whether the perimeter is drawn.

b) FILL COLOR: determines the color in which the interior of a filled area is drawn. This applies only if INTERIOR STYLE is 'solid' or 'hatch'.

c) HATCH INDEX: determines which entry in the hatch table is used if 'hatch' interior style is selected.

d) PATTERN INDEX: determines which entry in the pattern table is used if 'pattern' interior style is selected.

e) PERIMETER TYPE: determines the type of the line to be used to draw the perimeter.

f) PERIMETER WIDTH: determines the width of the perimeter.

g) PERIMETER COLOR: determines the color in which the perimeter is drawn.

4.6.4 TEXT Bundle. The TEXT BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) TEXT FONT INDEX: determines the style of the graphical display of the text characters.

b) TEXT PRECISION: determines the fidelity with which characters must be displayed and positioned.

c) CHARACTER EXPANSION FACTOR: determines the deviation of the width/height ratio of character from the ratio indicated by the font designer.

d) CHARACTER SPACING: determines the amount of blank space added between characters in a string.

e) TEXT COLOR: determines the color in which the text characters are drawn.

4.6.5 Specification Modes. The VDM provides the mechanism for both 'absolute' and 'scaled' specification of the modal values of the size-related elements (LINE WIDTH, MARKER SIZE, and PERIMETER WIDTH). 'Absolute' specification means that the sizes are given in VDC units. 'Scaled' specification means that the size is specified as a scale factor to be applied at metafile interpretation time to the device-dependent nominal size for the associated primitive.

4.6.6 TEXT Attributes. The representation and placement of text characters on a device is controlled by the attributes TEXT FONT INDEX, CHARACTER SET INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOR, and CHARACTER HEIGHT and by the control element LOCAL BACKGROUND COLOR. The placement and orientation of text strings is controlled by the attributes CHARACTER ORIENTATION, CHARACTER PATH, and TEXT ALIGNMENT. TEXT BUNDLE INDEX is an index into the text bundle table, each entry of which contains values for the bundleable attributes. Although the placement and size of text can be precisely specified by the attributes mentioned, the fidelity of rendering depends on the current TEXT PRECISION.

The choice of character font (that is, the style of the characters to be displayed) is determined independently of the character set. However, the specified font must be related to the character set being used in order for the font to have some meaning. Roman and Gothic are examples of commonly used fonts for Latin-based alphabets.

The attributes in the character representation and placement group (above) and TEXT BUNDLE INDEX may be changed within a string. The TEXT element is tagged to show it is not complete and provides the first portion of the string. The TEXT element may be followed by the desired text attribute element(s) and then by an APPEND TEXT element, which provides the next portion of the string. This may be repeated as often as necessary, with the final APPEND TEXT tagged to indicate that the string is complete. Note that a metafile interpreter generally cannot display any of the text until the string is complete because of TEXT ALIGNMENT and the way in which attribute changes affect the definition of the text extent rectangle (see below). Text may be displayed before the string is complete only in the following cases:

Path	Vertical Alignment	Horizontal Alignment
right	normal vertical or baseline	normal horizontal, left, or continuous (0,0)
left	normal vertical or baseline	normal horizontal, right, or continuous (1,0)
down	top, capline, normal vertical, or continuous (0,1)	normal horizontal or center
up	baseline, bottom, normal vertical, or continuous (0,0)	normal horizontal or center

Selection of characters from different character sets within a string is done by the CHARACTER SET INDEX element. The assignment of meaning to the index values of character set index is done with the Metafile Descriptor element CHARACTER SET LIST. Control codes (such as SI, SO, and ESC) are permitted within the text string, but their meaning is not standardized and should be used to access character sets only by prior agreement between the metafile generator and the metafile interpreter.

Selection of fonts from different font tables is done by the TEXT FONT INDEX element. The assignment of meaning to the index values of TEXT FONT INDEX is done with the Metafile Descriptor element FONT LIST.

The font coordinate system is illustrated in Figure 4-2. The character body encloses all of the drawn parts of all characters in the font (that is, no descender extends lower than 'bottom', and no accent mark or oversized symbol extends higher than 'top'). The left and right edges of the character body may be defined on a per-character basis to accommodate variable widths, proportional spacing, and kerning. The body exceeds the actual character symbol width and height as necessary to provide

adequate white space between characters, such that text is readable and adequately separated when adjacent character bodies are flush (that is, when CHARACTER SPACING is 0). The character body is defined in this way to permit alignment of multiline text without overlaps in the metafile environment.

The CHARACTER HEIGHT specifies the VDC distance between the capline and baseline of the font (see Figure 4-2). The CHARACTER EXPANSION FACTOR specifies the deviation of the width to height ratio of the characters from the ratio indicated by the font designer (see Figure 4-3). CHARACTER SPACING specifies how much additional space is to be inserted between two adjacent character bodies (see Figure 4-4). If the value of CHARACTER SPACING is zero, the character bodies are arranged one after the other along the TEXT PATH with only the intercharacter spacing designated by the font designer. If the value of CHARACTER SPACING is positive, additional space is inserted between character bodies. If the value of CHARACTER SPACING is negative, adjacent character bodies overlap although the character symbols themselves may not. Character spacing is specified as a fraction of the CHARACTER HEIGHT.

CHARACTER ORIENTATION specifies the character up vector and base vector, which fix the orientation, skew, and distortion of the characters, and also determine the sense of 'right', 'left', 'up,' and 'down' for TEXT PATH and TEXT ALIGNMENT (see Figure 4-5).

The way in which software above the metafile generator and/or the metafile generator itself may use CHARACTER ORIENTATION is described. To generate the CHARACTER ORIENTATION and CHARACTER HEIGHT elements, a vector whose length is the character height (baseline-to-capline) and whose direction is the desired character up vector is created. A second vector is also created with the same length, whose direction is negative 90-degrees from the up vector. This pair of vectors may be transformed before being given to the metafile generator as the parameters to CHARACTER ORIENTATION. The length of the transformed up vector may then be used to generate the CHARACTER HEIGHT element. If an anisotropic transformation is in effect, the character height must be respecified by the metafile generator for each change in orientation (see Figure 4-6). The CHARACTER HEIGHT and CHARACTER ORIENTATION are decoupled to permit changing character height (but not orientation) within a string. Thus, to the metafile interpreter, the absolute lengths of the vectors in CHARACTER ORIENTATION are not significant; only their directions and the ratio of their lengths are significant.

CHARACTER PATH has the possible values 'right', 'left', 'up', and 'down'. It specifies the writing direction of the text string.

'Right' means in the direction of the character base vector.
'Left' means 180-degrees from the character base vector.
'Up' means in the direction of the character up vector.
'Down' means 180-degrees from the character up vector.

For the 'up' and 'down' text path directions, the characters are arranged so that the centers of the character bodies are on a straight line in the direction of the CHARACTER UP VECTOR. For the 'left' and 'right' text path directions, the characters are arranged so that the baselines of the characters are on a straight line parallel to the direction of the character base vector. These composition rules also hold true when characters of different heights, expansion factors, or fonts are intermixed in a string by means of attribute changes between nonfinal TEXT elements and subsequent APPEND TEXT elements.

Alignment of text is done with respect to a text extent rectangle, which is derived by joining the character bodies of the characters in the string according to the current status of the attributes and the composition rules described.

For CHARACTER PATH = 'left' or 'right',

TOPLINE: the topline farthest from the baseline
CAPLINE: the capline farthest from the baseline
HALFLINE: the halfline farthest from the baseline
BOTTOMLINE: the bottomline farthest from the baseline
LEFT: leftmost edge of leftmost character body
RIGHT: rightmost edge of rightmost character body
CENTER: halfway between left and right edges

FOR CHARACTER PATH = 'up' or 'down',

TOPLINE: topline of topmost character
CAPLINE: capline of topmost character
HALFLINE: halfway between halflines of topmost and bottommost character
BASELINE: baseline of bottommost character
BOTTOMLINE: bottomline of bottommost character
LEFT: the left edge farthest from the center line
RIGHT: the right edge farthest from the center line

See Figure 4-7. Note that the relationship of topline to capline, bottomline to baseline, and the placement of the halfline are font-dependent. It is for this reason that the various defining lines of the text extent rectangle need not be derived from the same character body. This is a function of the text height, text font and character expansion factor changes within a string.

The TEXT ALIGNMENT attribute controls the positioning of the text extent rectangle in relation to the text position (see Figure 4-8).

The horizontal component of TEXT ALIGNMENT has five possible values: 'left', 'center', 'right', 'normal horizontal' and 'continuous horizontal'. If the *horizontal component* is 'left', the left side of the text extent rectangle passes through the text position. Similarly, if the value is 'right', the right side of the text extent rectangle passes through the text position. If the horizontal component is 'center', the text position lies midway between the left and right sides of the text extent rectangle. In this case, if TEXT PATH = 'up' or 'down', the straight line passing through the centerlines of the characters also passes through the text position. The vertical component of TEXT ALIGNMENT has seven possible values: 'top', 'cap', 'half', 'base', 'bottom', 'normal vertical', and 'continuous vertical'. A vertical alignment value of 'top', 'cap', 'half', 'base', or 'bottom' causes the text to be moved such that the corresponding defining line of the text extent rectangle passes through the text position. For both horizontal and vertical alignment, normal values are converted to the appropiate value, as indicated in Section 5, at text element *elaboration time and thereafter treated as above*.

If the value of TEXT ALIGNMENT is 'continuous horizontal', an additional value, 'horizontal alignment' (a real number normalized so that 1.0 corresponds to the width of the text extent rectangle) is used as an offset from the text position to the left side of the text extent rectangle (see Figure 4-9).

If the value of TEXT ALIGNMENT is 'continuous vertical', an additional value, 'vertical alignment' (a real number normalized so the 1 (unity) corresponds to the height of the text extent rectangle) is used as an offset from the text position to the bottom side of the text extent rectangle.

The foregoing examples have been illustrated for the case of the character up vector and the character base vector being orthogonal. When they are not, the text extent rectangle becomes a parallelogram, with the sides remaining parallel to the two orientation vectors. The centerline skews to remain parallel with the left and right edges of the text extent parallelogram. The height of the text extent rectangle is measured along the skewed edge (not perpendicular to the baseline), and the distance to be moved for alignment is done along the angle made by the appropriate orientation vector (see Figure 4-10). Right is in the direction of the character base vector, and left is in the opposite direction.

The continuous values of alignment are necessary for the VDM generator to ensure that, when displaying rows of text, the ascenders of one row and the descenders of the next do not overlap. Since inquiry of the dimensions of the text extent rectangle cannot be provided at metafile generation time, other means have been provided to align more than one row of characters to the same text position. This means that the alignment parameters must be able to exceed the maximum dimensions of the text extent rectangle. Use of a continuous parameter allows specification of interrow space, analogous to intercharacter space.

As an example of the set of the continuous alignment attributes, consider the display of four rows of left-justified text, each consisting of a single string specified by a single TEXT element. To ensure that ascenders and descenders do not interfere between rows and that, in addition, there is a space of at least one-half the maximum size of a character between the descenders of one row with raised accent marks or oversized symbols of another row, TEXT ALIGNMENT should be set to ('left', 'continuous vertical'), and 'vertical alignment' to 0.0. Then, output the first row with the text position equal to the lower-left corner of the string. Now, set 'vertical alignment' to 1.5, and output the second string with the same text position. This places the second row below the first because of the change in alignment. The last two rows are output in the same way with 'vertical alignment' set to 3.0 and 4.5, and the text position parameters to TEXT unchanged. A value of 1.0 assures no overlap between rows; anything greater than 1.0 guarantees additional non-printing space.

4.6.7 Color Attributes. The VDM uses the RGB additive color model used in many video devices and in color television.

The VDM provides two mechanisms for color selection: 'direct' and 'indexed'. In 'direct' color selection, the color is defined by providing values for the normalized weights of the RGB components. In 'indexed' color selection, the color is defined by an index into a table of direct color values. Selection of one of these mechanisms may be done by an element in each picture descriptor.

For direct color specification, color values are a three-tuple of values providing the normalized weight of the red, green, and blue components of the desired color. In the abstract, each component of the three-tuple is a continuous range. Thus, for the red component, one end of the range indicates that no red is included, and the other end indicates that the maximum possible red is included in the color with an infinite number of red component values in between. However, the precision with which the components are specified is limited by device and application dependencies and by the VDM encoding. Typically, the encodable precision exceeds the precision required by the application and

available from a device. If the metafile was generated using more precise color specification than is available with the device, the display of the picture is implementation dependent. If the metafile was generated with less precise color specification than available to the device, the lower precision specifications are mapped into the higher precision by normalizing the values within the more precise range.

Figure 4-11 gives an example of this mapping where the VDM generator intends color specification with three bits of precision, and the device provides four bits of precision. The example shows only the mapping for the red component of color--green and blue map exactly the same way. The precision of three bits effectively states that there are eight levels of red possible--evenly spaced from no red component to full red. The precision of four bits says there are 16 levels provided by the device. The mapping must preserve full intensity and zero intensity of the components. Note that this discussion of direct color specification also applies to the direct color values used to set color table entries.

4.6.8 Fill Area Attributes. Separate control is provided over the appearances of the interior and the perimeter of filled-area primitives. In particular, the perimeter can be either visible or invisible. If visible, the individual perimeter attribute or the fill-area bundle index (according to the perimeter ASF values) govern the appearance. The "realized perimeter" is defined to be the zero-width ideal boundary line of the filled-area if the perimeter is invisible, and the finite-width displayed line if the perimeter is visible (this line is centered on the ideal line). The discussion of interior in the remainder of this standard should be considered to pertain to "realized interior". Realized interior is defined as extending into and terminating at the realized perimeter.

4.7 Escape Elements

VDM ESCAPE elements describe device- or system-dependent data in the VDM. VDM ESCAPEs may be included in the metafile at the discretion of the user, but direct effects and side effects of the use of nonstandardized elements are beyond the scope of this standard. This standard imposes no constraints on the functional intent or content of data passed by the VDM ESCAPE mechanism.

4.8 External Elements

External elements communicate information not directly related to the generation of a graphical image. They may appear anywhere in the VDM.

The MESSAGE element specifies a string of characters used to communicate information to operators at VDM interpretation time. This element is intended to be used to provide special device-dependent information necessary to process a VDM. Control over the position and appearance of the character string is not provided.

The APPLICATION DATA element is intended to allow applications to store and access private (presumably nongraphical) data. This data is not intended to be interpreted or processed by a VDM interpreter, but is available to be returned directly to the application.

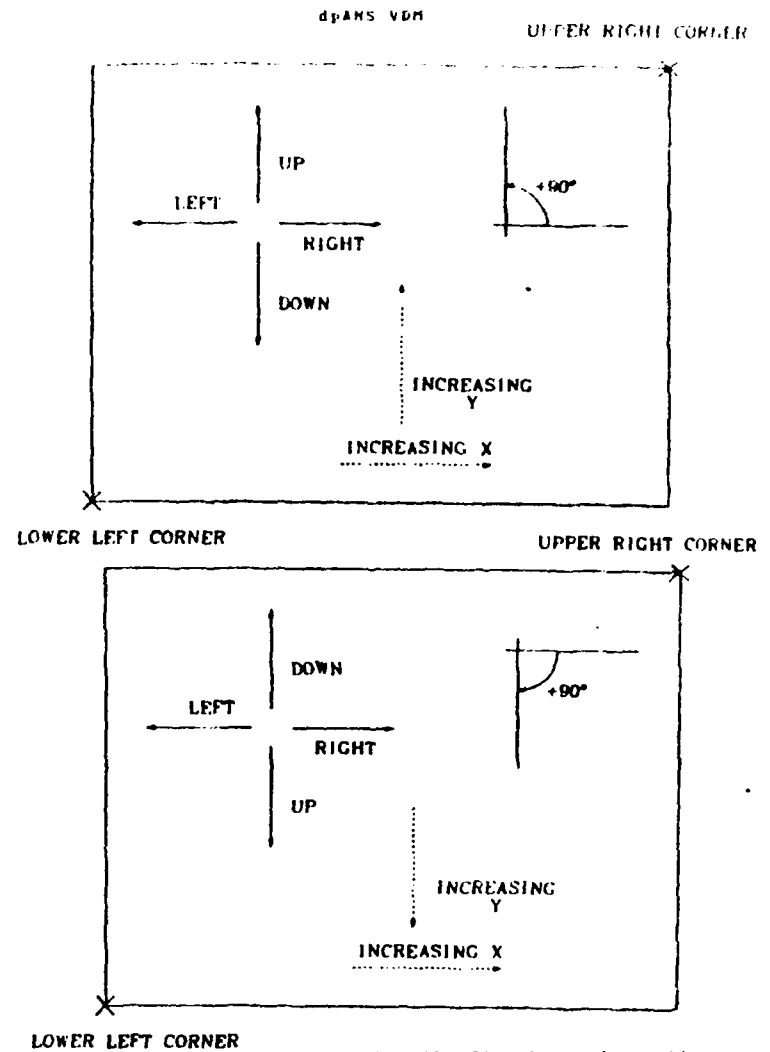


Figure 4-1: VDC EXTENT establishes the direction of positive and negative angles.

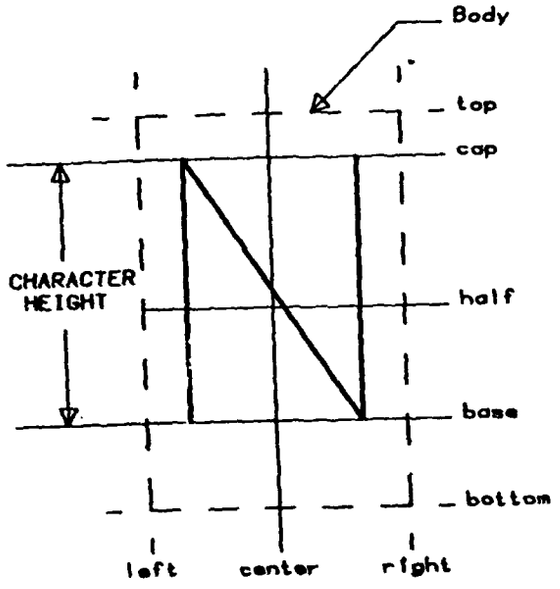


Figure 4-2: Font description coordinate system.

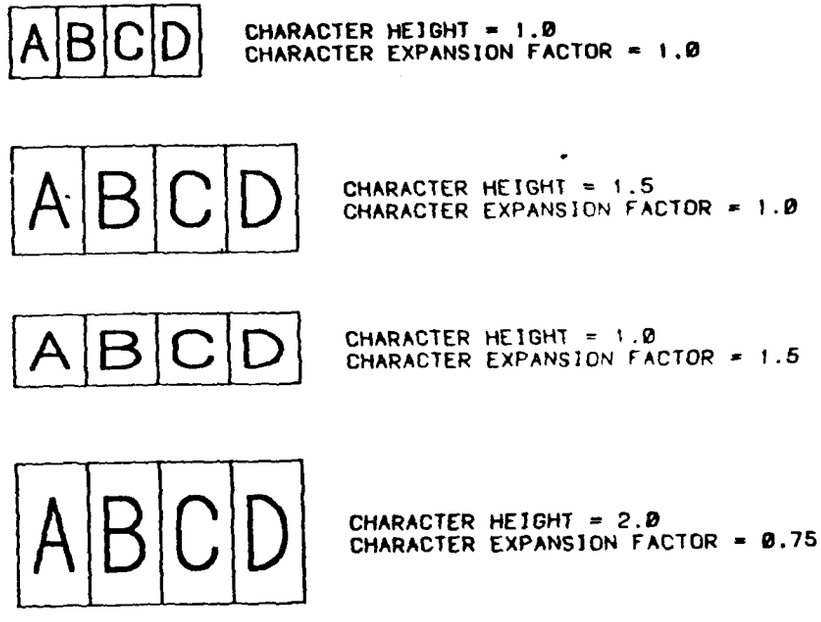
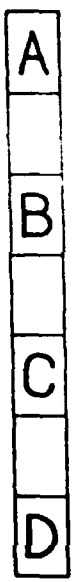


Figure 4-3: CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR.

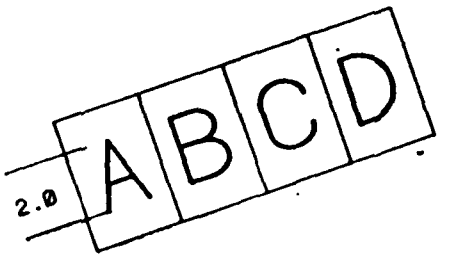
CHARACTER HEIGHT = 1.0
CHARACTER SPACING = 0.67
CHARACTER PATH = right

CHARACTER HEIGHT = 1.0
CHARACTER SPACING = -0.67
CHARACTER PATH = right

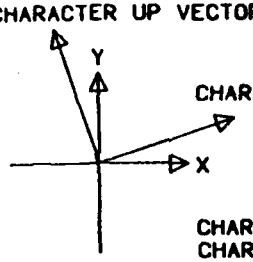


CHARACTER HEIGHT = 1.0
CHARACTER SPACING = 2.0
CHARACTER PATH = down

Figure 4-4: CHARACTER SPACING.



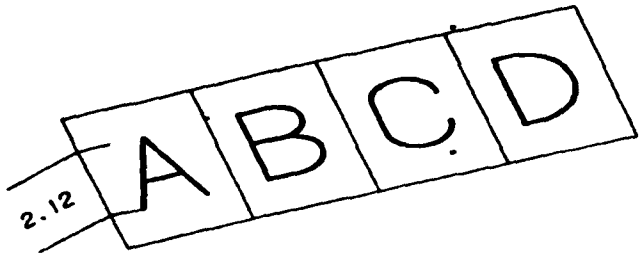
CHARACTER UP VECTOR = (-1,3)



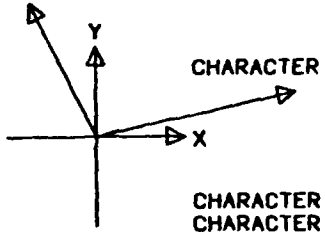
CHARACTER BASE VECTOR = (3,1)

CHARACTER ORIENTATION = (-1,3,3,1)
CHARACTER PATH = right
CHARACTER HEIGHT = 2.0

Figure 4-5: CHARACTER ORIENTATION.



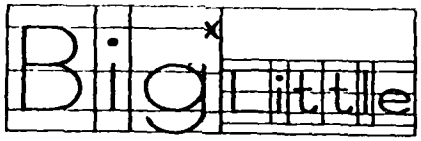
CHARACTER UP VECTOR = (-1.5,3)



CHARACTER BASE VECTOR = (4.5,1)

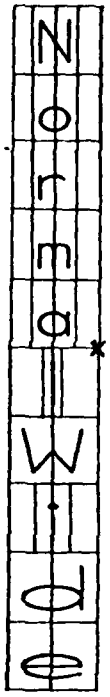
CHARACTER ORIENTATION = (-1.5,3,4.5,1)
CHARACTER PATH = right
CHARACTER HEIGHT = 2.12

Figure 4-6: CHARACTER HEIGHT and CHARACTER ORIENTATION after anisotropic transformation.



TEXT ALIGNMENT = (center,cap,0,0)

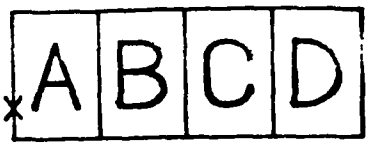
CHARACTER PATH = right
CHARACTER HEIGHT = 2.0
String = Big
CHARACTER HEIGHT = 1.0
Appended String = Little



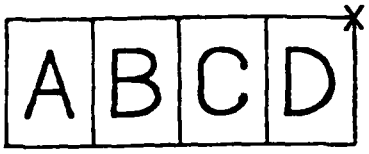
TEXT ALIGNMENT = (right,center,0,0)

CHARACTER PATH = down
CHARACTER HEIGHT = 1.0
CHARACTER EXPANSION FACTOR = 1.0
String = Normal
CHARACTER EXPANSION FACTOR = 2.0
Appended String = Wide

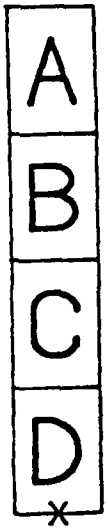
Figure 4-7: Discrete text alignment with appended text and proportional spacing.



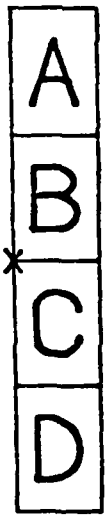
TEXT ALIGNMENT = (left,base,0,0)
CHARACTER PATH = right



TEXT ALIGNMENT = (right,top,0,0)
CHARACTER PATH = right



TEXT ALIGNMENT = (center,bottom,0,0)
CHARACTER PATH = down



TEXT ALIGNMENT = (left,half,0,0)
CHARACTER PATH = down

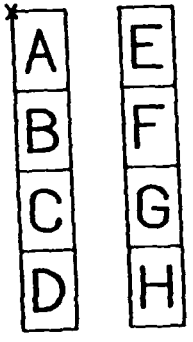
Figure 4-8: Discrete text alignment.

TEXT ALIGNMENT = (continuous horizontal,base,0.25,0)
CHARACTER PATH = right

TEXT ALIGNMENT = (continuous horizontal,continuous vertical, -0.25,-0.25)
CHARACTER PATH = right

String 1 = ABCD
TEXT ALIGNMENT = (left,continuous vertical,0,1)
CHARACTER PATH = right

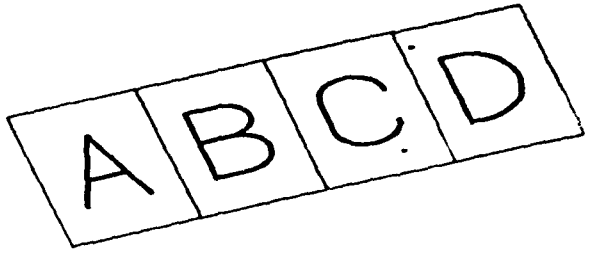
String 2 = EFGH
TEXT ALIGNMENT = (left,continuous vertical,0,2.5)
CHARACTER PATH = right



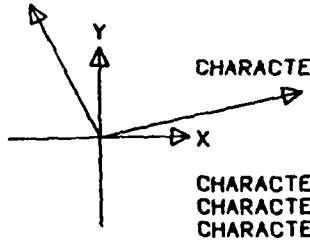
String 1 = ABCD
TEXT ALIGNMENT = (continuous horizontal,top,0,0)
CHARACTER PATH = down

String 2 = EFGH
TEXT ALIGNMENT = (continuous horizontal,top,2.0,0)
CHARACTER PATH = down

Figure 4-9: Continuous text alignment.



CHARACTER UP VECTOR = (-1.5,3)



CHARACTER BASE VECTOR = (4.5,1)

CHARACTER ORIENTATION = (-1.5,3,4.5,1)
CHARACTER PATH = right
CHARACTER HEIGHT = 2.12

TEXT ALIGNMENT = (left,continuous vertical, 0,2.0)

Figure 4-10: Continuous text alignment after anisotropic transformation.

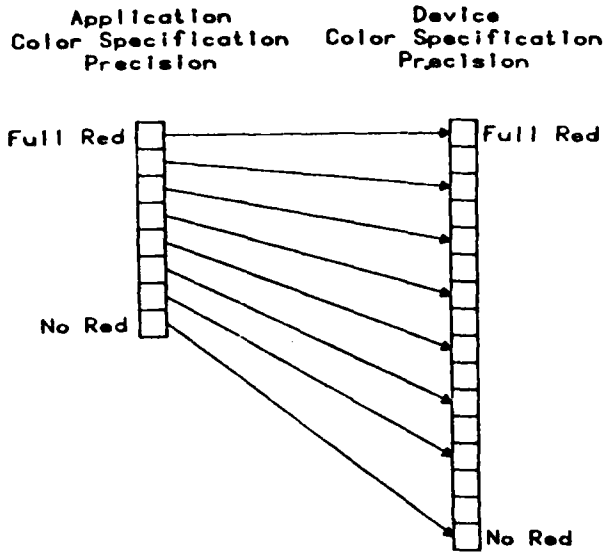


Figure 4-11: Mapping of direct color specification from lower to higher precision.

5. VIRTUAL DEVICE METAFILE ELEMENTS

5.1 Introduction

The metafile elements are discussed in this chapter.

The Metafile Descriptor Elements (Section 5.2) describe the functional content, default conditions, identification, and characteristics of the VDM.

The Control Elements (Section 5.3) specify metafile delimiters, address space, picture delimiters, and format descriptions of the VDM elements.

The Picture Descriptor Elements (Section 5.4) set the interpretation modes of attribute elements for the entire picture.

The Graphical Elements (Section 5.5) describe geometric objects in the VDM.

The Attribute Elements (Section 5.6) describe the appearance of graphical elements.

The VDM Escape Elements (Section 5.7) describe device- and system-dependent elements used to construct a picture, but are otherwise not standardized.

The External Elements (Section 5.8) communicate information not directly related to the generation of a graphical image.

The format used throughout this chapter to define the Metafile element set is designed to separate functionality from coding. Each element is named, the parameters are described, data types are listed, and a description of implicit relationships is added to clarify how the element fits into the system.

The order in which parameters will occur in a parameter list is not to be assumed from the order in which they are mentioned in this section, but is deferred to the description of specific encodings.

Notation:

Data Types		Meaning
CI	Color Index	Integer pointer into a table of color values.
CD	Color Direct	Three tuple of red, green, blue color values.
E	Enumerated	Set of standardized values. The set is defined by enumerating the identifiers that denote the values.
I	Integer	Number with no fractional part.
IX	Index	Integer pointer into a table of values, or integer used to select from among a set of enumerated values.
P	Point	Two VDC values representing the x and y coordinates of a point in VDC space.
R	Real	Number with integer and fractional portion, only one of which need exist.
S	String	Sequence of characters.
VDC	VDC value	Single real or integer values (as determined by VDC TYPE) in VDC space.

Type IX parameters used as enumeration selectors in some elements have a fixed number of values with defined and standardized meanings, and have other values available for implementation-dependent definition and use. It is anticipated that the standardized values will be expanded in future versions of the VDM. To avoid possible conflict with user-defined values, the standardized and user-available values are assigned to distinct ranges of the IX parameter. Negative values of IX are to be allocated for user-defined meanings, and nonnegative values are reserved for (future) standardization.

Combinations of simple types can also be used where n is an unspecified number (for example, nP or 2R,2I). Also, lists of types can be expressed (for example, I,E,R,E).

How these data types are represented in a given encoding of the VDM is specified in Parts 2 and 3.

5.2 Metafile Descriptor Elements

5.2.1 VDM VERSION

Parameters:

version (I)

Description:
The metafile conforms to the specified version of the VDM standard. This element must occur in the Metafile Descriptor of every metafile. Subsequent versions of the metafile will use higher numbered versions.

Related Elements:
None

Discussion:
This element specifies to which metafile standard the contents of the metafile conform. This element anticipates future revisions of the metafile standard.

References:

5.2.2 VDM DESCRIPTION

Parameters:

description (S)

Description:
The contents of the metafile are described in a nonstandardized way by this entry.

Related Elements:
None

Discussion:
This element allows the VDM to be identified with descriptive text such as author, place of origin, etc.

References:

5.2.3 VDC TYPE

Parameters:

An enumerative value that represents the data type (integer or real) of the virtual device coordinates (E)

Description:
None

Related Elements:
INTEGER VDC PRECISION
REAL VDC PRECISION

Discussion:
None

References:
4.2.2

5.2.4 NON-VDC INTEGER PRECISION

Parameters:

The exact form of the parameter depends on the specific encoding.

Description:
The precision for operands of data type integer (I) is specified for subsequent data of type I. The precision is defined as the field width measured in units applicable to the specific encoding.

Related Elements:
None

Discussion:
None

References:
4.2.2

5.2.5 NON-VDC REAL PRECISION

Parameters:

The exact form of the parameter(s) depend(s) on the specific encoding.

Description:
The precision for operands of data type real (R) is specified for subsequent data of type R. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameters that define subfields of data type R.

Related Elements:
None

Discussion:
None

References:
4.2.2

5.2.6 INDEX PRECISION

Parameters:

The exact form of the parameter depends on the specific encoding.

Description:
The precision for operands of data type index (IX) is specified for subsequent data of type IX. The precision is defined as the field width measured in units applicable to the specific encoding.

Related Elements:
None

Discussion:
None

References:
4.2.2

5.2.7 COLOR PRECISION

Parameters:

The exact form of the parameter depends on the specific encoding.

Description:
The precision for operands of data type color direct (CD) is specified for subsequent data of type CD. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameter(s) that define subfield(s) of data type CD.

Related Elements:
None

Discussion
None

References:
4.2.2

5.2.8 COLOR INDEX PRECISION

Parameters:

The exact form of the parameter depends on the specific encoding.

Description:
The precision for operands of data type color index (CI) is specified for subsequent data of type CI. The precision is defined as the field width measured in units applicable to the specific encoding.

Related Elements:
None

Discussion:
None

References:
4.2.2

5.2.9 MAXIMUM COLOR INDEX

Parameters:

maximum color index (CI)

Description:
The parameter represents an upper bound (not necessarily the least upper bound) on the color index values that may be encountered in the metafile.

Related Elements:
COLOR TABLE
LINE COLOR
MARKER COLOR
FILL COLOR
PERIMETER COLOR
CELL ARRAY
LOCAL BACKGROUND COLOR

Discussion:
Metafile interpreters are informed of the maximum color table facilities that may be required to process the metafile.

References:
4.2.2

5.2.10 VDM ELEMENT LIST

Parameters:

elements (E)

Description:
All of the elements that may be encountered in the metafile are listed. VDM ELEMENT LIST must occur in the Metafile Descriptor of every metafile.

Related Elements:
VDM DEFAULTS REPLACEMENT

Discussion:
This information can be used by the interpreters to determine the maximum facilities necessary for interpreting the VDM. The list represents an upper bound of functional capability. It need not be the least upper bound. Every element in the VDM must be in the list, but the list may include elements not found in the VDM.

References:

5.2.11 VDM DEFAULTS REPLACEMENT

Parameters:

Control and attribute element list.

Description:
Each element in the element list will have the same format, meaning, and parameter data types as it does when it occurs outside the VDM DEFAULTS REPLACEMENT element. Section 6 gives default values for those VDM elements for which defaults make sense. Substitute or replacement values for the defaults may be defined with the VDM DEFAULTS REPLACEMENT. Any subset of the elements listed in Section 6 may be included. Each picture within this metafile assumes that at BEGIN PICTURE the model values of all elements are the default values whether the defaults are the Section 6 values or come from this element.

Related Elements:
VDM ELEMENT LIST
BEGIN PICTURE

Discussion:
The content and format for elements in the default list are the same as the content and format for setting corresponding elements. The exact format of the parameter list is not further elaborated here in order to allow freedom for bindings to treat this complex element in the manner best suited to the bindings.

References:
4.2.2

5.2.12 FONT LIST

Parameters:

font names (nS)

Description:
This element permits selection of named fonts to be accessed with the TEXT FONT INDEX. The first font defined in the font list is assigned to index 1. The second to index 2, etc.

The strings may contain the names of type faces registered with some agency such as the United States Copyright Office, or private names. The former is recommended for metafile transportability, because registration ensures unique naming of fonts.

Related Elements:
TEXT
APPEND TEXT
TEXT FONT INDEX

Discussion:
None

References:
4.2.2

5.2.13 CHARACTER SET LIST

Parameters:
list of:
character set type - (one of: 94-character sets, 96-character sets, 94-character multibyte sets, 96-character multibyte sets, non-ISO 2022 coding system character sets), designation sequence tail n[E:S].

Description:
This element defines the list of character sets to be accessed with the CHARACTER SET INDEX. The first set defined in the list is assigned to index 1, the second to index 2, etc. Each set designation consists of an enumerated parameter indicating its type and a string that is formed by taking the escape sequence used to designate this character set and deleting the escape character (1/11) and the intermediate character indicating the G-set to which the set is assigned. The registry of the aforementioned escape sequences is the "ISO International Register of Character Sets to be used with Escape Sequences", published by ECMA (European Computer Manufacturers Association).

Related Elements:
CHARACTER SET INDEX

Discussion
For example, to assign ASCII to character set index 1, the character set type would be 94-character. The standard escape sequence to designate ASCII as the G0 character set is ESC 2/8 4/2 where 2/8 is the intermediate character indicating that the G0 set is being assigned. The first 2 bit combinations, ESC and 2/8 are ignored, and the designation sequence tail parameter consists of the bit combination 4/2.

Information regarding the designation sequence tail parameter can be found in the "ISO International Register of Character Sets to be Used with Escape Sequences". The Secretariat for the register is the European Computer Manufacturers Association (ECMA), Rue du Rhone 114, CH-1204, Geneva, Switzerland.

References:
4.2.2

5.3 *Control Elements*

5.3.1 BEGIN METAFILE

Parameters:

identifier (S)

Description:
This is the first element of the Metafile.

Related Elements:
END METAFILE

Discussion:
The parameter allows multiple VDMs to be recorded and addressed on the same output medium. Nesting of one metafile within another is not allowed. The identifier parameter is available for use by metafile generators and interpreters in a manner that is not further standardized.

References:
4.3

5.3.2 END METAFILE

Parameters:

None

Description:
This is the last element of the Metafile.

Related Elements:
BEGIN METAFILE

Discussion:
None

Reference:

5.3.3 BEGIN PICTURE

Parameters:

on-off flag (one of: on,off) (E)
identifier (S)

view surface color specifier, either
view surface color index (if the VDM default color selection mode is 'indexed') (CI)

or

view surface color value (red, green, blue) (if the VDM default color selection mode is 'direct') (CD)

Description:
This is the first element of a picture. It forces all elements to return to the default values. If a new picture begins with a cleared view surface, the initial color of the view surface is the color specified by the color index or value, or is the device-dependent default view surface color if the on-off flag is 'off'.

Related Elements:
BEGIN PICTURE BODY
END PICTURE
VDM DEFAULTS REPLACEMENT

Discussion:
BEGIN PICTURE and END PICTURE bound the set of elements of a single picture in the VDM. Every picture in a metafile is totally independent from every other picture and always starts with a BEGIN PICTURE. This independence is enforced by returning the modal values of all elements to their default values at the start of the picture. The identifier parameter is available for use by metafile generators and interpreters in a manner that is not further standardized.

Each picture defines a graphical image independent of the other pictures. As suggested in Appendix D, presentation of each picture on a cleared view surface is the most expected picture. Because view surface clearing is not standardized, interpreters are free to *compose* images by overlaying pictures.

References:
4.3

5.3.4 BEGIN PICTURE BODY

Parameters:

None

Description:
This element demarks the end of the Picture Descriptor and the beginning of the body of the picture. It thus informs the metafile interpreter of the transition from the Picture Descriptor to the graphical attribute, and control elements that define the picture.

Related Elements:
BEGIN PICTURE END PICTURE

Discussion:
None

References:
4.3

5.3.5 END PICTURE

Parameters:

None

Description:
This is the last element of a picture.

Related Elements:
BEGIN PICTURE BEGIN PICTURE BODY

Discussion:
This is a symmetrical element to BEGIN PICTURE. No explicit actions are specified to occur when this element is encountered. Only external and escape elements may occur between *END PICTURE* and BEGIN PICTURE or between END PICTURE and *END METAFILE*.

References:
4.3

5.3.6 LOCAL BACKGROUND COLOR

Parameters:

on-off switch (one of: on, off) (E)

background color specifier, either
local background color index (if the color selection mode is 'indexed') (CI)

or

local background color value (red, green, blue) (if the color selection mode is 'direct') (CD)

Description:
This function sets the local background color index or value, and the background color on-off switch.

Some devices, particularly raster scan devices, may use this background color for special purposes. When the on-off switch is 'on', the following primitives may be affected.

(1) POLYLINE: When LINE TYPE is nonsolid, lines may alternate between the current LINE COLOR and the LOCAL BACKGROUND COLOR.

(2) PERIMETER: When PERIMETER TYPE is nonsolid, perimeters may alternate between the current perimeter color and the LOCAL BACKGROUND COLOR.

(3) POLYMARKER: For devices that display markers within raster cells, pixels that are not the marker definition may be displayed in the LOCAL BACKGROUND COLOR.

(4) TEXT (APPEND TEXT): For devices that display TEXT within raster cells, pixels that are not the character definition may be displayed in the LOCAL BACKGROUND COLOR.

(5) INTERIOR: when INTERIOR STYLE is hatch, pixels that are not the hatch pattern may be displayed in the LOCAL BACKGROUND COLOR.

Turning the on-off switch to 'off' in such devices will cause 'transparent' to be substituted for 'background color' if the device is capable of making the substitution. For example, a raster scan device that draws text in character boxes and that can either color all noncharacter pixels in the character box or make them transparent, will color the pixels in the background color if the on-off switch is on and will make them transparent if the switch is off. This element does not cause the view surface to be repainted to the background color.

Related Elements:
POLYLINE
TEXT
APPEND TEXT
POLYMARKER
POLYGON
CIRCLE
ARC
ARC CLOSE

Discussion:
None

References:
None

5.3.7 INTEGER VDC PRECISION

Parameters:

The exact form of the parameter depends on the specific encoding.

Description:
The indicated precision for operands of data type point (P) and operands of data type VDC value (VDC) are specified for subsequent data of type P and of type VDC. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameters that defined subfields of data types P and VDC.

Related Elements:
VDC TYPE

Discussion:
This element enables metafiles to change the form of parameters in other metafile elements in the middle of a picture so that more efficient storage of data can be used when less precision is required. The exact form of the parameter depends on the specific encoding (binding).

References:
None

5.3.8 REAL VDC PRECISION

Parameters:

The exact form of the parameter(s) depend(s) on the specific encoding.

Description:
The indicated precision for operands of data type point (P) and operands of data type VDC value (VDC) are specified for subsequent data of type P and of type VDC. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameters that define subfields of data type P and VDC.

Related Elements:
VDC TYPE

Discussion:
This element enables metafiles to change the form of parameters in other metafile elements in the middle of a picture so that more efficient storage of data can be used when less precision is required. The exact form of the parameter depends on the specific encoding (binding).

References:

5.3.9 VDC EXTENT

Parameters:

first corner (P)
second corner (P)

Description:
The two corners define a rectangular extent in VDC space that is the "region of interest" for the succeeding VDM elements.

Related Elements:
SCALING MODE

Discussion:
The first corner represents the lower-left corner of the picture, and the second corner represents the upper-right corner of the picture as seen by the viewer of the picture. The values of the coordinates for any dimension may be either increasing or decreasing from the first to the second corner. For example, for devices with an upper-left origin, a picture may be described in coordinates that map directly to the device but still may be displayed correctly on a device with a lower-left origin.

The VDC EXTENT thus establishes the sense and orientation of VDC space (that is, the directions of the positive x (+x) and positive y (+y) axes, and whether the +y axis is 90-degrees clockwise or 90-degrees counterclockwise from the +x axis). See Section 4.3.2 and Figure 4-1.

In particular, VDC EXTENT establishes the direction of positive and negative angles as follows: positive 90-degrees is defined to be the right angle from the positive x-axis to the positive y-axis.

Note that some attributes such as text attributes (for example, the directions of the up and base component vectors of CHARACTER ORIENTATION and, therefore, the meaning of the enumerative values 'right', 'left', 'up', 'down') are intimately bound to these definitions.

Specification of values outside VDC EXTENT is permitted in VDM elements. It is intended that the visible portion of an image be contained within VDC EXTENT. It thus is intended to demark the region of interest in a picture.

References:

5.3.10 CLIP RECTANGLE

Parameters:

xmin,xmax,ymin,ymax (4VDC)

Description:
Parameters xmin, xmax, ymin, and ymax define a rectangular extent in VDC space, which defines the clipping rectangle.

When CLIP INDICATOR is 'on', only the portions of graphics elements inside or on the boundary of the clip rectangle are drawn.

Related Elements:
VDC EXTENT
CLIP INDICATOR

Discussion:
This element is necessary to provide local clipping areas within VDC space. For example, high-quality text could be sent to the Metafile as a text string and clip rectangle so that the interpreter can clip character strokes.

Note that the method of specifying the rectangular region for CLIP RECTANGLE is xmin, xmax, ymin, ymax, whereas the method for VDC EXTENT is the lower-left and upper-right corner points. The latter specification, with no restrictions on the numerical order of the x and y components of the two points, is intended to allow setting the orientation of the VDC space. The min-max method for CLIP RECTANGLE is chosen to avoid the possibility of inversion or orientation being implied for the clipping rectangle.

References:

5.3.11 CLIP INDICATOR

Parameters:

indicator (flag) (one of: on, off) (E)

Description:
When CLIP INDICATOR is 'off', clipping of graphical elements is not required.

When CLIP INDICATOR is 'on', only those portions of graphical entities within or on the boundary of the clipping rectangle are meant to be drawn. Actions outside the clip rectangle are legal but are meant to have no visible *effect* outside the clip rectangle.

Related Elements:
VDC EXTENT
CLIP RECTANGLE

Discussion:
Only when CLIP INDICATOR is 'on' is clipping of graphical elements mandated. It is implementation and interpreter dependent whether or not clipping is done to some limit such as VDC EXTENT or display surface boundaries even when CLIP INDICATOR is 'off'. Such action is not precluded by this standard, and may be handled by the interpreter in accord with the particular needs of the implementation and driven device(s).

References:

5.4 Picture Descriptor Elements

5.4.1 SCALING MODE

Parameters:

scaling mode (one of: abstract, metric) (E)
metric scale factor (R)

Description:
The scaling mode parameter defines the meaning of the VDC. If set to 'abstract', the VDC space is dimensionless and the picture is correctly displayed at any size: the metric scale factor parameter is ignored. If set to 'metric', the VDC space has implied measure: one VDC unit represents one millimeter multiplied by the metric scale factor. In this case the picture is correctly displayed at the indicated size only. If used, SCALING MODE must appear in the Picture Descriptor, after BEGIN PICTURE and before BEGIN PICTURE BODY.

Related Elements:
VDC EXTENT

Discussion:
Metric scaling mode provides a device-independent means of generating output at a known scale factor. In metric mode, a scale factor of unity implies that coordinates in the associated picture(s) are in units of millimeters; a scale factor of 25.4 would imply coordinates in inches. Scale factors less than one may be used to achieve good physical resolution while storing coordinates in integer form. The movement desired on the physical view surface equals the movement in VDC units multiplied by the metric scale factor.

References:
4.4.1

5.4.2 COLOR SELECTION MODE

Parameters:

color selection mode (one of: indexed, direct) (E)

Description:
Two methods of color selection are supported: by color table entries ('indexed') or by red, green, and blue color values ('direct').

Only one color mode may be used within a picture. The mode may be defaulted or explicitly set with the COLOR SELECTION MODE element. All occurrences of color-setting elements (LOCAL BACKGROUND COLOR, LINE COLOR, MARKER COLOR, FILL COLOR, PERIMETER COLOR, TEXT COLOR) must be in the current mode as well as the color list of CELL ARRAY. COLOR SELECTION MODE must appear in the Picture Descriptor, after BEGIN PICTURE and before BEGIN PICTURE BODY.

Related Elements:
LINE COLOR
MARKER COLOR
FILL COLOR
PATTERN TABLE
TEXT COLOR
LOCAL BACKGROUND COLOR
PERIMETER COLOR
CELL ARRAY

Discussion:
None

References:
4.4.2

5.4.3 LINE WIDTH SPECIFICATION MODE

Parameters:

line width specification mode (one of: absolute, scaled)

Description:
Two methods of directly specifying line width are supported: absolute measure in VDC ('absolute'), or a scaling factor to be applied to the *device-dependent* nominal line width at metafile interpretation time.

Only one line width mode may *be used within a picture*. The mode may be defaulted or may be set explicitly with the LINE WIDTH SPECIFICATION MODE element. If used, LINE WIDTH SPECIFICATION MODE must be in the *Picture Descriptor*, after the BEGIN PICTURE element and before any graphical or attribute elements. All occurrences of line width elements must have parameters in the current mode.

Related Elements:
LINE WIDTH

Discussion:
None

References:
4.4.3

5.4.4 MARKER SIZE SPECIFICATION MODE

Parameters:

marker size specification mode (one of: absolute, scaled)

Description:
Two methods of directly specifying marker size are supported: absolute measure in VDC ('absolute'), or a scaling factor to be applied to the device-dependent nominal marker size at metafile interpretation time.

Only one marker size mode may be used within a picture. The mode may be defaulted or may be set explicitly with the MARKER SIZE SPECIFICATION MODE element. If used, MARKER SIZE SPECIFICATION MODE must be in the Picture Descriptor after the BEGIN PICTURE element and before BEGIN PICTURE BODY. All occurrences of marker size elements must have *parameters in the current mode*.

Related Elements:
MARKER SIZE

Discussion:
None

References:
4.4.3

5.4.5 PERIMETER WIDTH SPECIFICATION MODE

Parameters:

perimeter width specification mode (one of: absolute, scaled)

Description:
Two methods of directly specifying perimeter width are supported: absolute measure in VDC ('absolute'), or a scaling factor to be applied to the device-dependent nominal perimeter width at metafile interpretation time.

Only one perimeter width mode may be used within a picture. The mode may be defaulted or may be set explicitly with the PERIMETER WIDTH SPECIFICATION MODE element. If used, PERIMETER WIDTH SPECIFICATION MODE must be in the Picture Descriptor after the BEGIN PICTURE element and before BEGIN PICTURE BODY. All occurrences of perimeter width elements must have parameters in the current mode.

Related Elements:
PERIMETER WIDTH

Discussion:
None

References:
4.4.3

5.5 Graphical Elements

5.5.1 POLYLINE

Parameters:

point list (nP)

Description:
A line is drawn from the first point in the parameter list to the second point, from the second point to the next point, ..., and from the next-to-last point to the last point. The interpretation of a zero-length line segment is implementation dependent.

Related Elements:
POLYLINE BUNDLE INDEX
SET ASPECT SOURCE FLAGS
LINE TYPE
LINE WIDTH
LINE COLOR
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.5

5.5.2 POLYMARKER

Parameters:

point list (nP)

Description:
The marker corresponding to the currently selected marker type is drawn at each of the points in the point list. If the marker type is one of the five predefined markers, it is drawn centered at each of the points. Other implementation-dependent markers may have other alignments where desired. If the resulting marker is completely within the clipping area, the entire marker is drawn. If the marker position is outside the clipping rectangle, nothing is displayed. If the marker position is within the clipping rectangle but any part of the marker is outside the clipping area, the result is device or interpreter dependent.

Related Elements:
POLYMARKER BUNDLE INDEX
SET ASPECT SOURCE FLAGS
MARKER TYPE
MARKER SIZE
MARKER COLOR
LOCAL BACKGROUND COLOR

Discussion:
None

Reference:
4.5

5.5.3 POLYGON

Parameters:

point list (nP)

Description:
A boundary of a polygonal region is defined by connecting each vertex to its successor in the ordered point list and connecting the last vertex to the first. The polygonal region may be nonsimple. For example, edges are allowed to cross. In this way, subareas can be created. Any given point is considered inside the polygon if a straight line from the given point to infinity intersects the polygon edges an odd number of times. If this line passes through a vertex point tangentially, the intersection count is not changed.

A nondegenerate polygon (one with three or more vertices, not all of which are colinear) is displayed with interior as defined by the FILL AREA BUNDLE INDEX, SET ASPECT SOURCE FLAGS, and interior style attributes. The appearance of the boundary is controlled by the 'perimeter visibility' component of INTERIOR STYLE, by the perimeter attributes, and by LOCAL BACKGROUND COLOR.

The interpretation of degenerate polygons is implementation dependent. Some recommendations are provided in Appendix D.

Related Elements:
PERIMETER TYPE
PERIMETER WIDTH
PERIMETER COLOR
HATCH INDEX
PATTERN INDEX
PATTERN TABLE
FILL AREA BUNDLE INDEX
SET ASPECT SOURCE FLAGS
INTERIOR STYLE
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.5

5.5.4 CIRCLE

Parameters:

point (P)
radius (VDC)

Description:
A circle of the specified radius at the specified VDC position is displayed with interior as defined by the FILL AREA BUNDLE INDEX, FILL AREA ASF, and interior style attributes. The appearance of the boundary is controlled by the 'perimeter visibility' component of INTERIOR STYLE, by the perimeter attributes, and by LOCAL BACKGROUND COLOR. The interpretation of a circle with zero or negative radius is implementation dependent. Some recommendations are provided in Appendix D.

Related Elements:
PERIMETER TYPE
PERIMETER WIDTH
PERIMETER COLOR
FILL AREA BUNDLE INDEX
MATCH INDEX
PATTERN INDEX
PATTERN TABLE
SET ASPECT SOURCE FLAGS
INTERIOR STYLE
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.5

5.5.5 ARC

Parameters:

starting point, intermediate point, ending point (3P)

Description:
A circular arc is displayed from the starting point, through the specified intermediate point, to the specified ending point.

If the three specified coordinates result in only one or two distinct points, or if the three coordinates are colinear, the interpretation of this element is implementation dependent. Some recommendations are provided in Appendix D.

Related Elements:
POLYLINE BUNDLE INDEX
SET ASPECT SOURCE FLAGS
LINE TYPE
LINE WIDTH
LINE COLOR
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.5

5.5.6 ARC CLOSE

Parameters:

starting point, intermediate point, ending point (3P)
close type (one of: pie, chord) (E)

Description:

A circular arc is displayed from the specified starting point through the specified intermediate point, to the specified ending point. The close types are illustrated in figure 5-1.

If close type is 'chord', the segment defined by the arc and the chord from the starting point to the ending point is displayed with interior as defined by the FILL AREA BUNDLE INDEX, FILL AREA ASF, and interior style attributes. The appearance of the boundary is controlled by the 'perimeter visibility' component of INTERIOR STYLE and the perimeter attributes.

If close type is 'pie', the pie sector defined by the computed arc center, the specified starting point, and the ending point is displayed with interior as defined by the FILL AREA BUNDLE INDEX, FILL AREA ASF, and the interior style attributes. The appearance of the boundary is controlled by the 'perimeter visibility' component of INTERIOR STYLE, the perimeter attributes, and LOCAL BACKGROUND COLOR.

If the three specified coordinates result in only one or two distinct points, or if the three coordinates are colinear, the interpretation of this element is implementation dependent. Some recommendations are provided in Appendix D.

Related Elements:

PERIMETER TYPE
PERIMETER WIDTH
PERIMETER COLOR
FILL AREA BUNDLE INDEX
HATCH INDEX
PATTERN INDEX
PATTERN TABLE
SET ASPECT SOURCE FLAGS
INTERIOR STYLE
LOCAL BACKGROUND COLOR

Discussion

None

References:

4.5

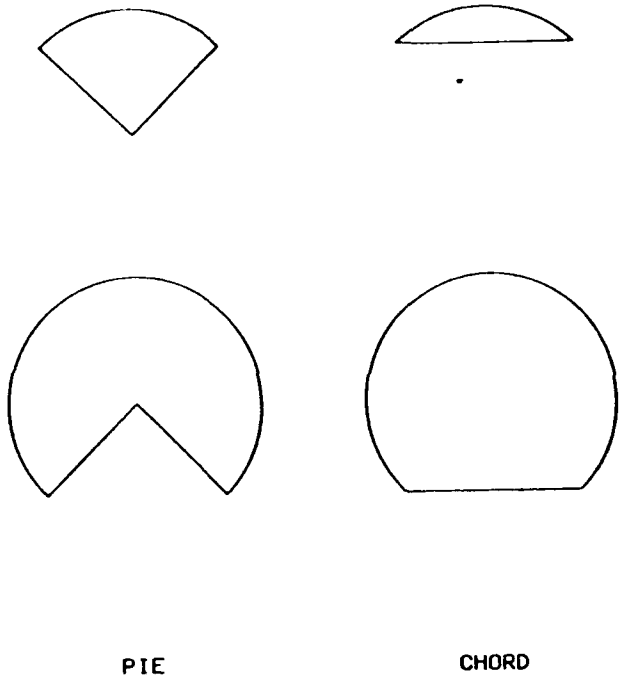


Figure 5-1: ARC CLOSE specifications with 'pie' and 'chord'.

5.5.7 TEXT

Parameters:

point (P)
flag (one of: final, not final) (E)
string (S)

Description:

The character codes specified in the string are interpreted to obtain the associated symbols from the currently selected character set. Characters are displayed on the view surface as specified by the text attributes. Control characters (such as CR, LF, SO or SI) in a string are allowed, but have no standardized effect.

The characters are dimensioned according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of the character placement in the string relative to CHARACTER ORIENTATION is according to CHARACTER PATH.

The flag parameter is used to permit changing the following text attributes within a string which will be aligned as a single block: TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOR, CHARACTER HEIGHT, CHARACTER SET INDEX, LOCAL BACKGROUND COLOR, and TEXT BUNDLE INDEX. If the flag is set to 'not final', the character codes in the string parameter are accumulated, along with the current attribute setting. In this case, only the attribute setting elements listed above are allowed between this element and the APPEND TEXT element. With the exception of the VDM ESCAPE element, no other metafile elements of any type are allowed. VDM ESCAPE is permitted but has no standardized effect.

VDM ESCAPE permits device-dependent or implementation-dependent changes within strings (for example, underlining, which is difficult to do after-the-fact with proportionally-spaced fonts). Proper care must be taken by the metafile writer that only the appropriate escapes be used in this situation.

If the flag is set to 'final', the string parameter constitutes the entire string to be displayed. The position of the string relative to the text point parameter is according to TEXT ALIGNMENT. Text elements with a null string parameter are legal and may be followed by the allowed text attributes and APPEND TEXT as described above.

Related Elements:

CHARACTER SET INDEX
APPEND TEXT
TEXT BUNDLE INDEX
SET ASPECT SOURCE FLAGS
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOR
CHARACTER HEIGHT
CHARACTER ORIENTATION
CHARACTER PATH
TEXT ALIGNMENT
CHARACTER SET LIST
FONT LIST
LOCAL BACKGROUND COLOR

Discussion:

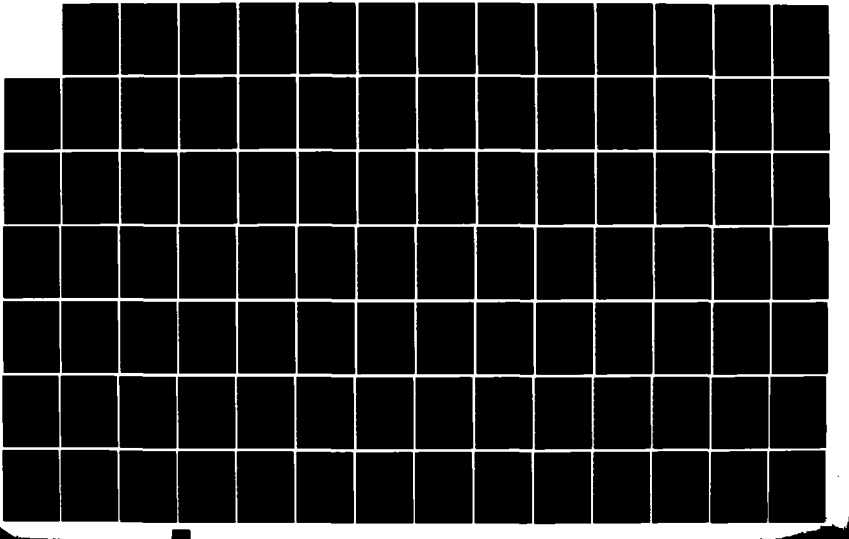
None

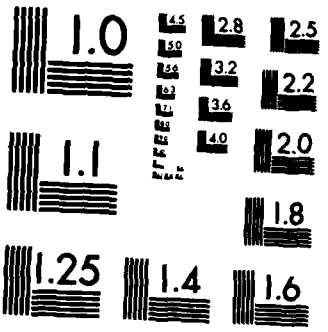
References:

4 5

AD-A153 322 STUDY OF RASTER METAFILE FORMATS(U) BATTELLE COLUMBUS LABS OH M R TAYLOR ET AL. JAN 85 ETL-0363 DAAG29-81-D-0100 2/3

UNCLASSIFIED F/G 9/2 NL





5.5.8 APPEND TEXT

Parameters:

flag (one of: final, not final) (E)
string (S)

Description:

The character codes specified in the string are appended to the *string defined* by *preceding* nonfinal TEXT and APPEND TEXT elements. The codes are interpreted to obtain the associated symbols from the current character set. Characters are displayed on the view surface as specified by the text attributes. Control characters (such as CR, LF, SO, or SI) in a string are allowed but have no standardized effect.

The characters are dimensioned according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of the character placement in the string relative to CHARACTER ORIENTATION is according to CHARACTER PATH.

The flag parameter is used to permit changing the text attributes within a string which will be aligned as a single block: TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOR, CHARACTER HEIGHT, CHARACTER SET INDEX, LOCAL BACKGROUND COLOR, and TEXT BUNDLE INDEX. If the flag is set to 'not final', the character codes in the string parameter are accumulated, along with the current attribute setting. In this case, only the attribute setting elements listed above are allowed between this element and the APPEND TEXT element. With the exception of the external element VDM ESCAPE, no other metafile elements of any type are allowed.

VDM ESCAPE permits *device-dependent* or implementation-dependent changes within strings (for example, underlining, *which* is difficult to do after-the-fact with proportionally spaced fonts). Proper care must be taken by the metafile writer that only the appropriate escapes be used in this situation. VDM ESCAPE is permitted but has no standardized effect.

If the flag is set to 'final', the accumulated string parameter constitutes the entire string to be displayed. APPEND TEXT elements with a null string parameter are legal and may be followed by the allowed text attributes and further APPEND TEXT elements as described above.

Related Elements:

CHARACTER SET INDEX
TEXT
TEXT BUNDLE INDEX
SET ASPECT SOURCE FLAGS
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOR
CHARACTER HEIGHT
CHARACTER ORIENTATION
CHARACTER PATH
TEXT ALIGNMENT
CHARACTER SET LIST
FONT LIST
LOCAL BACKGROUND COLOR

Discussion: None

References:

4.5

5.5.9 CELL ARRAY

Parameters:

3 corner points P, Q, and R (3P)

dx,dy (2I)

cell color specifiers, either
color indexes (if the color selection mode is 'indexed') (dx*dy*CI)

or

color values (red value, green value, blue value) (if the color selection is 'direct') (dx*dy*CB)

Description:

In the general case, P,Q,R can delimit an arbitrary parallelogram. P and Q delimit the end points of a diagonal of the parallelogram, and R defines a third corner.

In the simplest case, the three corner points, P,Q,R, define a rectangular area in VDC space. This area is subdivided into dx*dy contiguous rectangles as follows. The edge from P to R is subdivided into dx equal intervals, and the edge from R to Q is subdivided into dy equal intervals. The grid implied consists of dx*dy identical cells. The color list *consists of dx*dy color* specifications, conceptually an array of dimensions dx and dy representing respectively the column and row dimensions. Array element (1,1) is mapped to the cell at corner P, and array element (dx,1) is mapped to the cell at corner R. Array element (dx,dy) is mapped to the cell at corner Q. Hence, the color elements are mapped within rows running from P to R, and with the rows incrementing in order from R to Q.

Figure 5-2 illustrates this in the case of the common left-to-right, top-to-bottom color mapping order of many devices.

Related Elements:
COLOR SELECTION MODE
COLOR TABLE

Discussion:
None

References:
4.5

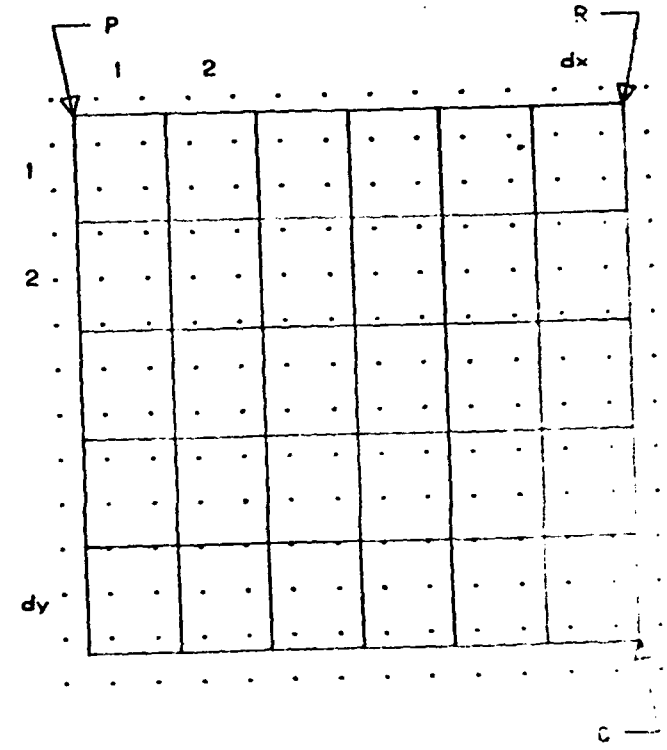


Figure 5-2: *dx by dy* rectangle mapped onto display surface. Lines indicate cell array locations. Dots indicate pixels.

5.6 Attribute Elements

5.6.1 SET ASPECT SOURCE FLAGS

Parameters:

list of: pairs of
ASF type, ASF value (individual, bundled) m[IX,E]

Description:
The designated Aspect Source Flags (ASFs) are set to the values indicated by the parameter. The following ASF types are assigned:

0 - line type ASF
1 - line width ASF
2 - line color ASF
3 - marker type ASF
4 - marker size ASF
5 - marker color ASF
6 - interior style ASF
7 - hatch index ASF
8 - pattern index ASF
9 - fill color ASF
10 - perimeter type ASF
11 - perimeter width ASF
12 - perimeter color ASF
13 - text font index ASF
14 - text precision ASF
15 - character expansion factor ASF
16 - character spacing ASF
17 - text color ASF

Related Elements:
LINE TYPE
LINE WIDTH
LINE COLOR
MARKER TYPE
MARKER SIZE
MARKER COLOR
INTERIOR STYLE
FILL COLOR
HATCH INDEX
PATTERN INDEX
PERIMETER TYPE
PERIMETER WIDTH
PERIMETER COLOR
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOR

TEXT BUNDLE INDEX
FILL AREA BUNDLE INDEX
POLYMARKER BUNDLE INDEX
POLYLINE BUNDLE INDEX

Discussion:
The Aspect Source Flags determine the attribute values that will be bound to a primitive. If the ASF for a particular aspect of a primitive is set to 'individual', the value used is the value of the corresponding individually specified attribute of the primitive. If the ASF is set to 'bundled', the value used is the value of the corresponding aspect of the bundle pointed to by the current bundle index for the primitive.

Changing the value of an ASF within a picture is legal but will have no retroactive effect on any previous graphical element.

References:
4.6

5.6.2 POLYLINE BUNDLE INDEX

Parameters:

polyline bundle index (IX)

Description:
The polyline bundle index is set to the value specified by the parameter. When subsequent POLYLINE or ARC elements occur, the values for LINE TYPE, LINE WIDTH, and LINE COLOR are taken from the corresponding components of the indexed bundle if the ASFs for those attributes are set to 'bundled'.

If the ASF for a given attribute is 'individual', this element does not affect the value used for that attribute until the ASF returns to 'bundled'.

Legal values are positive integers.

Related Elements:
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.1

5.6.3 LINE TYPE

Parameters:

line type indicator (IX)

Description:
The line type indicator is set to the value specified by the parameter.

When the LINE TYPE ASF is 'individual', subsequent POLYLINE elements are displayed with this line type.

When the LINE TYPE ASF is 'bundled', this element does not affect the display of subsequent POLYLINE elements until the ASF returns to 'individual'.

The following line types are assigned:

0: solid
1: dash
2: dot
3: dash dot
4: dash dot dot

Nonnegative values of the index are reserved for standardized line types, and negative values are available for implementation-dependent use.

Related Elements:
POLYLINE
ARC
LOCAL BACKGROUND COLOR
SET ASPECT SOURCE FLAGS

Discussion:
Line type is intended to be maintained continuously across the interior vertices within a single POLYLINE element. Continuity between separate, but graphically connected, POLYLINE elements is not mandated by this standard; nor is continuity mandated across interior sections of a single POLYLINE element that may have been clipped away.

References:
4.6.1

5.6.4 LINE WIDTH

Parameters:

line width specifier, either
absolute line width
(if line width specification mode is 'absolute') (VDC)

or

line width scale factor
(if line width specification mode is 'scaled') (R)

Description:
The absolute line width or line width scale factor is set as specified by the parameter.

When the LINE WIDTH ASF is 'individual', subsequent elements are displayed according to the size specification of this element.

When the LINE WIDTH ASF is 'bundled', this element does not affect the display of subsequent elements until the ASF returns to 'individual'.

Related Elements:
POLYLINE
ARC
SET ASPECT SOURCE FLAGS

Discussion:
The line width is measured perpendicular to the defining line (that is, it is independent of the orientation of the defining line). A wide line is aligned with its mathematically zero-width defining line such that the distance between the defining line and either edge is half the line width. The appearance of wide lines at interior vertices or corners (that is, whether they are mitered, rounded, etc.) is not addressed by this standard.

References:
4.6.1

5.6.5 LINE COLOR

Parameters:

line color specifier, either
line color index
(if the color selection mode is 'indexed') (CI)

or

line color value (red value, green value, blue value)
(if the color selection mode is 'direct') (CD)

Description:
The line color index or line color value is set as specified by the parameter(s).

When the LINE COLOR ASF is 'individual', subsequent elements are drawn in this line color.

When the LINE COLOR ASF is 'bundled', this element does not affect the interpretation of subsequent elements until the ASF returns to 'individual'.

Refer to Section 5.6.32 for legal values of line color index.

Related Elements:
POLYLINE
ARC
COLOR SELECTION MODE
COLOR TABLE
SET ASPECTS SOURCE FLAGS
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.6.1
4.6.7

5.6.6 POLYMARKER BUNDLE INDEX

Parameters:

polymarker bundle index (IX)

Description:

The polymarker bundle index is set to the value specified by the parameter. When subsequent POLYMARKER elements occur, the values for MARKER TYPE, MARKER SIZE, and MARKER COLOR are taken from the corresponding components of the indexed bundle if the ASFs for those attributes are 'bundled'.

If the ASF for a given attribute is 'individual', this element does not affect the value used for that attribute until its ASF returns to 'bundled'.

Legal values of POLYMARKER BUNDLE INDEX are positive integers.

Related Elements:

SET ASPECT SOURCE FLAGS

Discussion:

None

References:

4.6.2

5.6.7 MARKER TYPE

Parameters:

marker type (IX)

Description:

The marker type is set to the value specified by the parameter.

When the MARKER TYPE ASF is 'individual', subsequent POLYMARKER elements are displayed with this marker type.

When the MARKER TYPE ASF is 'bundled', this element does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'individual'.

The following marker types are assigned:

0: dot (.)
1: plus (+)
2: asterisk (*)
3: circle (o)
4: cross (x)

The marker type 'dot' is intended always to be displayed as the smallest visible point on the display surface at metafile interpretation time. It is thus intended to behave as a polypoint element.

Nonnegative values of the index are reserved for standardized marker types, and negative values of the index are available for implementation-dependent use.

Related Elements:

POLYMARKER
SET ASPECT SOURCE FLAGS

Discussion:

None

References:

5.6.8 MARKER SIZE

Parameters:

marker size specifier, either
absolute marker size
(if marker size specification mode is 'absolute') (VDC)

or

marker size scale factor
(if marker size specification mode is 'scaled') (R)

Description:

The absolute marker size or marker size scale factor is set as specified by the parameter. If absolute, the specified size is the maximum extent of the marker.

When the MARKER SIZE ASF is 'individual', subsequent POLYMARKER elements are displayed according to the size specification of this element.

When the MARKER SIZE ASF is 'bundled', this element does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'individual'.

Related Elements:

POLYMARKER
SET ASPECT SOURCE FLAGS

Discussion:

None

References:

4.6.2

5.6.9 MARKER COLOR

Parameters:

marker color specifier, either
marker color index
(if the color selection mode is 'indexed') (CI)

or

marker color value (red value, green value, blue value)
(if the color selection mode is 'direct') (CD)

Description:

The marker color index or marker color value is set as specified by the parameter(s).

When the MARKER COLOR ASF is 'individual', subsequent POLYMARKER elements are displayed with this marker color.

When the MARKER COLOR ASF is 'bundled', this element does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'individual'.

Refer to Section 5.6.32 for legal values of marker color index.

Related Elements:

POLYMARKER
COLOR SELECTION MODE
COLOR TABLE
SET ASPECT SOURCE FLAGS
LOCAL BACKGROUND COLOR

Discussion:

None

References:

4.6.2
4.6.7

5.6.10 FILL AREA BUNDLE INDEX

Parameters:

fill area bundle index (IX)

Description:

The fill area bundle index is set to the value specified by the parameter. When subsequent POLYGON, CIRCLE, or ARC CLOSE elements occur, values for INTERIOR STYLE, FILL COLOR, HATCH INDEX, PATTERN INDEX, PERIMETER TYPE, PERIMETER WIDTH, and PERIMETER COLOR are taken from the corresponding components of the indexed bundle, if the ASFs for these attributes are set to 'bundled'. If the ASF for a given attribute is 'individual', this element does not affect the value used for that attribute until its ASF returns to 'bundled'.

Legal values of FILL AREA BUNDLE INDEX are positive integers.

Related Elements:

SET ASPECT SOURCE FLAGS

Discussion:

None

References:

4.6.3

5.6.11 INTERIOR STYLE

Parameters:

interior style (IX)

perimeter visibility (one of: off, on) (E)

Description:

The interior style of the POLYGON, CIRCLE, and ARC CLOSE elements is set to the value specified by the parameter.

Negative values are used for private (nonstandard) interior styles and nonnegative values are reserved for standardized interior styles. When the INTERIOR STYLE ASF is 'individual', POLYGON, CIRCLE, and ARC CLOSE elements are displayed with this interior style.

When the INTERIOR STYLE ASF is 'bundled', this element does not affect the display of POLYGON, CIRCLE, and ARC CLOSE elements until the ASF returns to 'individual'.

The following interior styles are assigned:

0: hollow
1: solid
2: hatch
3: pattern

When the perimeter visibility value is 'on', the perimeter is displayed using the FILL AREA bundle attributes or individually set attributes, depending on the values of the perimeter ASF's.

When the perimeter visibility value is 'off', the perimeter is not displayed.

The interior fill style is used to determine in what style the area is to be filled. (See Section 4.6.8 for discussion of extent of the interior and relationship of the interior to the perimeter.)

'hollow' - no filling.

'solid' - fill the interior using the fill color currently selected (either via FILL AREA BUNDLE or FILL COLOR depending on the corresponding FILL COLOR ASF).

'hatch' - fill the interior using the fill color and the hatch index currently selected (either via FILL AREA BUNDLE or HATCH INDEX, depending on the corresponding HATCH INDEX ASF).

'pattern' - fill the interior using the pattern index currently selected (either via FILL AREA BUNDLE or PATTERN INDEX, depending on the corresponding PATTERN INDEX ASF) as an index into the pattern table.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
FILL AREA BUNDLE INDEX
PERIMETER TYPE
PERIMETER WIDTH
PERIMETER COLOR
FILL COLOR
HATCH INDEX
PATTERN INDEX
SET ASPECT SOURCE FLAGS
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.6.3

5.6.12 FILL COLOR

Parameters:

fill color specifier, either
fill color index
(if the color selection mode is 'indexed') (CI)

or

fill color value (red value, green value, blue value)
(if the color selection mode is 'direct') (CD)

Description:
The fill color index or fill color value is set as specified by the parameter(s).

When the FILL COLOR ASF is 'individual', subsequent POLYGON, CIRCLE, and ARC CLOSE elements are filled with this color.

When the FILL COLOR ASF is 'bundled', this element does not affect the display of these subsequent elements until the ASF returns to 'individual'.

The fill color index is a pointer into the color table. The fill color attribute is only significant if INTERIOR STYLE is either 'solid' or 'hatch'.

See Section 5.6.32 for the legal values of fill color index.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
COLOR SELECTION MODE
COLOR TABLE
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.3

5.6.13 HATCH INDEX

Parameters:

hatch index (IX)

Description:
The hatch index is set to the value specified by the parameter.

When the HATCH INDEX ASF is 'individual' and the interior style is 'hatch', subsequent POLYGON, CIRCLE, and ARC CLOSE elements are displayed using this hatch index.

When the HATCH INDEX ASF is 'bundled', this element does not affect the display of these subsequent elements until the ASF returns to 'individual'.

The hatch index is a pointer into the hatch tables of the device. The fill color attribute determines the color of the hatch lines.

Legal values of HATCH INDEX are positive integers.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
FILL COLOR
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.3

5.6.14 PATTERN INDEX

Parameters:

pattern index (IX)

Description:
The pattern index is set to the value specified by the parameter.

When the PATTERN INDEX ASF is 'individual' and the interior style is 'pattern', subsequent POLYGON, CIRCLE, and ARC CLOSE elements are displayed using this pattern index.

When the PATTERN INDEX ASF is 'bundled', this element does not affect the display of these subsequent elements until the ASF returns to 'individual'.

The pattern index is a pointer into the pattern tables.

Legal values of PATTERN INDEX are positive integers.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
PATTERN TABLE
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.3

5.6.15 PATTERN TABLE

Parameters:

pattern table index (IX)
m,n (2I)
pattern color specifier, either
color list indexes (if the color selection mode is 'indexed') (m*nIX)

or

color list (red, green, and blue values) (if the color selection mode is 'direct') (m*nCD)

Description:

The m and n values define a horizontal by vertical m by n array into which color values are mapped. The array is loaded sequentially by rows starting with the upper-left (minimum x,maximum y) corner. The array is defined in untransformed space.

Legal values of the pattern table index parameter are positive integers.

Related Elements:

POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
PATTERN SIZE
PATTERN REFERENCE POINT
COLOR SELECTION MODE

Discussion:

None

References:

4.6.3

5.6.16 PATTERN REFERENCE POINT

Parameters:

reference point (P)

Description:

The pattern reference point is set to the value specified by the parameter when the currently selected interior style is 'pattern'. This value is used in conjunction with pattern size for displaying fill area primitives.

When the currently selected interior style is 'hatch', it is interpreter dependent if the pattern reference point is used when displaying fill area primitives.

The position of the start of the pattern (or hatch) is defined by the pattern reference point. The pattern is mapped onto the filled area by conceptually replicating it in directions parallel to the sides of the pattern box until the interior of the complete fill area is covered.

Related Elements:

POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
PATTERN TABLE
PATTERN INDEX
PATTERN SIZE

Discussion:

None

References:

4.6.3

5.6.17 PATTERN SIZE

Parameters:

deltax (VDC)
deltay (VDC)

Description:
The pattern size is set to the value specified by the parameters.

When INTERIOR STYLE is set to 'pattern', subsequent POLYGON, CIRCLE, and ARC CLOSE elements are displayed with this pattern size.

The parameters deltax and deltay specify the width and height of an arbitrary rectangle. This arbitrary rectangle is subdivided by a grid of extent deltax/m by deltay/n where m and n are as in the PATTERN TABLE element. Colors are mapped into the resulting m by n array, defining an area texture. Conceptually, one corner of this arbitrary rectangle is placed at the PATTERN REFERENCE POINT and the rectangle is replicated as necessary, horizontally and vertically about the display surface. The coincidence of this imposed pattern and the interior to which it is to be applied defines the interior style for the filled area element being executed.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
INTERIOR STYLE
PATTERN TABLE
PATTERN INDEX
PATTERN REFERENCE POINT

Discussion:
None

References:

5.6.18 PERIMETER TYPE

Parameters:

perimeter type indicator (IX)

Description:
The perimeter type indicator is set to the value specified by the parameter.

When the PERIMETER TYPE ASF is 'individual', and perimeter visibility is 'on'. The perimeters of POLYGON, CIRCLE, and ARC CLOSE are displayed with this perimeter line type.

When the PERIMETER TYPE ASF is 'bundled', this element does not affect the display of subsequent elements until the ASF returns to 'individual'.

Line type indicator and perimeter type indicator should have the same correspondence between type (for example, 3) and representation (for example, dash-dot).

The following perimeter types are assigned:

0: solid
1: dash
2: dot
3: dash dot
4: dash dot dot

Nonnegative values of the index are reserved for standardized perimeter types, and negative values are available for implementation dependent use.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
SET ASPECT SOURCE FLAGS

Discussion:
Perimeter type is intended to be maintained continuously across the interior vertices within a single fill area element. Continuity across perimeter sections that may have been clipped away is not mandated by the standard.

References:
4.6.3

5.6.19 PERIMETER WIDTH

Parameters:

perimeter width specifier, either
absolute perimeter width (if perimeter width specification mode is 'absolute') (VDC)

or

perimeter width scale factor (if perimeter width specification mode is 'scaled') (R)

Description:
The absolute perimeter width or perimeter width scale factor is set as specified by the parameter.

When the *PERIMETER WIDTH ASF* is 'individual' and perimeter visibility is 'on', the perimeter of POLYGON, CIRCLE, and ARC CLOSE are displayed with this width.

When the *PERIMETER WIDTH ASF* is 'bundled', this element does not affect the display of subsequent elements until the ASF returns to 'individual'.

Related Elements:
PERIMETER WIDTH SPECIFICATION MODE
POLYGON
CIRCLE
ARC CLOSE
SET ASPECT SOURCE FLAGS

Discussion:
When a perimeter line is displayed on the virtual device, the perimeter line width is measured perpendicular to the defining line (that is, it is independent of the orientation of the defining line). A perimeter line is aligned with its mathematically zero-width defining line such that the distance between the defining line and either edge is half the line width.

References:
4.6.3

5.6.20 PERIMETER COLOR

Parameters:

perimeter color specifier, either
perimeter color index
(if the color selection mode is 'indexed') (CI)

or

perimeter color value (red value,green value,blue value)
(if the color selection mode is 'direct') (CD)

Description:
The perimeter color index or perimeter color value is set as specified by the parameter(s).

When the PERIMETER COLOR ASF is 'individual' and perimeter visibility is 'on', the perimeter of POLYGON, CIRCLE, and ARC CLOSE are displayed with this color.

When the PERIMETER COLOR ASF is 'bundled', this element does not affect the interpretation of subsequent elements until the ASF returns to 'individual'.

Related Elements:
POLYGON
CIRCLE
ARC CLOSE
COLOR SELECTION MODE
COLOR TABLE
SET ASPECTS SOURCE FLAGS
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.6.3
4.6.7

5.6.21 CHARACTER SET INDEX

Parameters:

character set index (IX)

Description:
The specified character set from the table specified in the CHARACTER SET LIST descriptor element becomes the currently selected character set and is used for the subsequent mapping of character codes to character symbols.

Legal values of character set index parameter are positive integers.

Related Elements:
TEXT
APPEND TEXT
TEXT FONT INDEX
CHARACTER SET LIST

Discussion:
One use of this element is to switch among character sets for different languages.

References:
4.6.4

5.6.22 TEXT BUNDLE INDEX

Parameters:

text bundle index (IX)

Description:
The text bundle index is set to the value specified by the parameter. When subsequent TEXT or APPEND TEXT elements occur, the values for TEXT FONT, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, and TEXT COLOR are taken from the corresponding components of the indexed bundle if the ASFs for these attributes are set to 'bundled'.

If the ASF for a given attribute is 'individual', this element does not affect the value used for that attribute until the ASF returns to 'bundled'.

Legal values of the text bundle index parameter are positive integers.

Related Elements:
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.4

5.6.23 TEXT FONT INDEX

Parameters:

font index (integer into a font table indicating which character font is to be used) (IX)

Description:

The font index is set to the value specified by the parameter.

When the TEXT FONT INDEX ASF is 'individual', subsequent TEXT and APPEND TEXT elements are displayed with this font index.

When the TEXT FONT INDEX ASF is 'bundled', this element does not affect the display of subsequent TEXT and APPEND TEXT elements until the ASF returns to 'individual'.

Legal values of the font index parameter are positive integers.

Related Elements:

TEXT
APPEND TEXT
CHARACTER SET LIST
SET ASPECT SOURCE FLAGS
FONT LIST

Discussion:

Care must be taken to ensure that the selected character set and the current text font are compatible. The font index is used to select a font from the font list table defined in the metafile descriptor (or the default font list, if none was specified). Appendix D gives recommendations for interpreters to follow in the case that the currently selected character set cannot be rendered in the specified text font.

References:

4.6.4

5.6.24 TEXT PRECISION

Parameters:

text precision (one of: string, character, stroke) (E)

Description:

The text precision is set to the value specified by the parameter.

When the TEXT PRECISION ASF is 'individual', subsequent TEXT and APPEND TEXT elements are displayed with this text precision.

When the TEXT PRECISION ASF is 'bundled', this element does not affect the display of subsequent TEXT and APPEND TEXT elements until the ASF returns to 'individual'.

The accuracy of execution of TEXT attributes can be controlled by one of three values.

If 'string' precision is specified, only the starting position of subsequent text strings need be guaranteed, and the manner in which the string is clipped is implementation dependent.

If 'character' precision is specified, the metafile interpreter must guarantee that the starting position of each character satisfy the relevant text attributes, thus guaranteeing orientation and placement of the string; however, skew, orientation, and size of each character are not guaranteed. All characters of the string which lie completely inside or outside the clipping region are clipped as appropriate, but the effect of clipping on a character whose character box is intersected by the clipping boundary is implementation dependent.

If 'stroke' precision is specified, the metafile interpreter must guarantee that the placement, skew, orientation, and size of all characters satisfy all standardized text attributes. Characters are clipped to the geometric accuracy of the device.

Related Elements:

TEXT
APPEND TEXT
SET ASPECT SOURCE FLAGS

Discussion:

None

References:

4.6.4

5.6.25 CHARACTER EXPANSION FACTOR

Parameters:

character expansion factor (R)

Description:
The character expansion factor is set to the value specified by the parameter.

When the CHARACTER EXPANSION FACTOR ASF is 'individual', subsequent TEXT and APPEND TEXT elements are displayed with this character expansion factor.

When the CHARACTER EXPANSION FACTOR ASF is 'bundled', this element does not affect the display of subsequent TEXT and APPEND TEXT elements until the ASF returns to 'individual'.

The character expansion factor specifies the deviation of the width/height ratio of the character from the ratio indicated by the font designer.

Related Elements:
TEXT
APPEND TEXT
CHARACTER HEIGHT
TEXT ALIGNMENT
CHARACTER ORIENTATION
SET ASPECT SOURCE FLAGS

Discussion:
The character expansion factor is a scalar. The resulting character width is the product of the CHARACTER HEIGHT multiplied by the width/height ratio (a characteristic of each font and a quantity that can vary on a character-by-character basis) for the character multiplied by the CHARACTER EXPANSION FACTOR. The character width so derived is further scaled by multiplying it by the ratio of the length of the character base vector to the length of the character up vector.

References:
4.6.4

5.6.26 CHARACTER SPACING

Parameters:

character spacing (R)

Description:
The character spacing is set to the value specified by the parameter.

When the CHARACTER SPACING ASF is 'individual', subsequent TEXT and APPEND TEXT elements are displayed with this character spacing.

When the CHARACTER SPACING ASF is 'bundled', this element does not affect the display of subsequent TEXT and APPEND TEXT elements until the ASF returns to 'individual'.

The parameter represents the desired space to be added between character bodies of a text string, which is in addition to any intercharacter spacing provided by the font within the character's body. It is specified as a fraction of the current CHARACTER HEIGHT attribute. The space is added along the character path. A negative value implies that characters may overlap.

When CHARACTER PATH is right or left, the character spacing is scaled by the ratio of the length of the character base vector to the length of the character up vector.

Related Elements:
TEXT
APPEND TEXT
CHARACTER HEIGHT
CHARACTER ORIENTATION
TEXT ALIGNMENT
SET ASPECT SOURCE FLAGS

Discussion:
None

References:
4.6.4

5.6.27 TEXT COLOR

Parameters:

text color specifier, either
text color index (if the color selection mode is 'indexed') (CI)

or

text color value (red value, green value, blue value) (if the color selection mode is 'direct') (CD)

Description:
The text color index or text color value is set as specified by the parameter(s).

When the TEXT COLOR ASF is 'individual', subsequent TEXT and APPEND TEXT elements are displayed with this text color.

When the TEXT COLOR ASF is 'bundled', this element does not affect the display of subsequent TEXT and APPEND TEXT elements until the ASF returns to 'individual'.

The text color index is a pointer into the color table.

Related Elements:
TEXT
APPEND TEXT
COLOR SELECTION MODE
COLOR TABLE
SET ASPECT SOURCE FLAGS
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.6.4
4.6.7

5.6.28 CHARACTER HEIGHT

Parameters:

character height (VDC)

Description:
The character height is set to the value specified by the parameter. Subsequent TEXT and APPEND TEXT elements are displayed with this character height.

The parameter represents the desired height of the character body, from baseline to capline, in VDC units; it must be a positive number. It is measured along the character up vector. If the character orientation vectors are not orthogonal, this will not be the perpendicular distance between baseline and capline.

Related Elements:
TEXT
APPEND TEXT
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT ALIGNMENT
CHARACTER ORIENTATION

Discussion:
None

References:
4.6.4

5.6.29 CHARACTER ORIENTATION

Parameters:

x character up component (VDC)
y character up component (VDC)
x character base component (VDC)
y character base component (VDC)

Description
The two vectors define the orientation and skew of the character body in subsequent TEXT and APPEND TEXT elements. The character base vector determines the sense of 'right' and 'left' for TEXT PATH and TEXT ALIGNMENT. The ratio of the length of the base vector to the length of the up vector is used as a scaling factor for the CHARACTER *EXPANSION FACTOR* and *CHARACTER SPACING* elements.

Related Elements:
TEXT
APPEND TEXT
TEXT PATH
TEXT ALIGNMENT
CHARACTER SPACING
CHARACTER EXPANSION FACTOR

Discussion:
The way in which software above the metafile generator and/or the metafile generator may use this element as follows. A vector whose length is the character height (baseline-to-capline) and whose direction is the desired character up vector is created. A second vector is also created with the same length, whose direction is negative 90-degrees from the up vector. This pair of vectors may be transformed before being given to the metafile generator as the parameters to CHARACTER ORIENTATION. If the resultant vectors are not orthogonal, the text extent rectangle becomes a parallelogram, and the characters are skewed. If the vectors have different lengths, the characters will be distorted. If the base vector makes a positive angle with respect to the up vector, the characters will be mirror-imaged, and the sense of 'left' and 'right' will be reversed for TEXT PATH and TEXT ALIGNMENT. If the two vectors are coincident, 180-degrees, or either vector is equal to (0,0), the element is ignored.

References:

5.6.30 CHARACTER PATH

Parameters:

character path (one of: right, left, up, down) (E)

Description:
The character path is set to the value specified by the parameter. Subsequent TEXT and APPEND TEXT elements are displayed with this character path.

This function sets the value of the character path attribute, specifying the writing direction of a text string relative to the character up vector and character base vector. 'Right' means in the direction of the character base vector. 'Left' means 180 degrees from the character base vector. 'Up' means in the direction of the character up vector. 'Down' means 180 degrees from the character up vector.

Related Elements:
TEXT
APPEND TEXT
TEXT ALIGNMENT
CHARACTER ORIENTATION

Discussion:
None

References:
4.6.4

5.6.31 TEXT ALIGNMENT

Parameters:

horizontal alignment (one of: left, center, right, normal horizontal, continuous horizontal) (E)

vertical alignment (one of: top, cap, half, base, bottom, normal vertical, continuous vertical) (E)

continuous horizontal alignment (R)

continuous vertical alignment (R)

Description:

The text alignment is set to the value specified by the parameters. Subsequent text strings are displayed with this text alignment.

The horizontal alignment type parameter is an enumerated data type with the possible values shown above. If its value is 'continuous horizontal', the continuous horizontal alignment parameter (which is a fraction of the side of the text extent rectangle perpendicular to character up vector) becomes significant.

The vertical alignment type parameter is an enumerated data type with the possible values shown above. If the value is 'continuous vertical', the continuous vertical alignment parameter (which is a fraction of the side of the text extent rectangle parallel to character up vector) becomes significant.

The "normal" parameters are dependent on the character path at the time of the elaboration of the TEXT or APPEND TEXT elements.

PATH	NORMAL HORIZONTAL	NORMAL VERTICAL
RIGHT	LEFT	BASELINE
LEFT	RIGHT	BASELINE
UP	CENTER	BASELINE
DOWN	CENTER	TOP

The continuous horizontal and vertical parameters may exceed the range of 0 to 1 in order to align a string with a coordinate outside its text extent rectangle.

Related Elements:
TEXT
APPEND TEXT
CHARACTER PATH
CHARACTER ORIENTATION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
CHARACTER HEIGHT

Discussion:
None

References:

5.6.32 COLOR TABLE

Parameters:

starting index (CI)
color list (red, green, blue) (nCB)

Description:
The color list elements are loaded, in the order specified, into the consecutive locations in the color table beginning at the starting index. Only the specified color table entries are changed. The effect of changes in the color table on any existing graphical elements that use the effected indexes is not standardized.

Legal values of the color index are non-negative integers.

Related Elements:
CELL ARRAY
LINE COLOR
MARKER COLOR
FILL COLOR
PATTERN TABLE
TEXT COLOR
PERIMETER COLOR
LOCAL BACKGROUND COLOR

Discussion:
None

References:
4.6.7

5.7 Escape Elements

5.7.1 VDM ESCAPE

Parameters:

function identifier (I)
data list

Description:
This element has been deliberately underspecified. Its purpose is to allow use of device capabilities not specified by this standard. Software making use of the VDM ESCAPE element is less portable. The function identifier parameter specifies the particular escape function to be used for processing the data list. The values of valid function identifiers are determined by prior agreement between metafile generators and interpreters.

Related Elements:
None

Discussion:
None

References:
4.7

5.8 External Elements

5.8.1 MESSAGE

Parameters:

action required flag (one of yes, no) (E)
text (S)

Description:
The MESSAGE element specifies a string of characters used to communicate information to operators at Metafile interpretation time through a path separate from normal graphical output.

Related Elements:
None

Discussion:
If the action required flag parameter is 'yes', the metafile interpreter may need to pause to wait for an operator response. Because the message and an associated pause may be directed at a particular device, only the interpreter may determine if a pause is appropriate. Character set selection for MESSAGE is independent of that for the TEXT graphical output element and is not specified by this standard.

References:
4.8

5.8.2 APPLICATION DATA

Parameters:

identifier (S)
data list (S)

Description:
This element has no effect on the picture or its encoding.

Related Elements:
None

Discussion:
The application data element is included for the metafile generator and interpreter to supplement the information in the Metafile in an application-dependent way. The contents of the data list are nongraphical and may include such information as history data associated with pictures, description of algorithms used, etc.

The identifier parameter is available for use by the application in a manner that is not further standardized.

References:
4.8

6. METAFILE DEFAULTS

6.1 Default Values

This section contains the Metafile default values that are used for those default values not explicitly set in the VDM DEFAULTS REPLACEMENTS element. The default values of some elements are dependent upon the values of other elements (for example, default CHARACTER HEIGHT is dependent upon VDC EXTENT). In these cases, the default of the dependent element is tied to the default of the other element, whether the latter is as defined in the table below or is defined with the VDM DEFAULTS REPLACEMENT elements. The value of the dependent element does not, however, change when the value of the element upon which it depends is changed explicitly by a metafile element. Rather, the value of the dependent element remains unchanged, in its default state, until explicitly changed by the occurrence of the element.

VDM VERSION
default: 1

VDC TYPE
default: Integer

VDM ELEMENT LIST
default: n/a

VDM DEFAULTS REPLACEMENT
default: n/a

VDC EXTENT
default: lower left (0.,0.), upper right (1.,1.) for real; lower left (0,0), upper right (32767,32767) for integer

CLIP RECTANGLE
default: VDC EXTENT

CLIP INDICATOR
default: off

SCALING MODE
default: abstract

METRIC SCALE FACTOR
default: 1.0

CHARACTER SET INDEX
default: 1

CHARACTER SET LIST
default: ASCII (first entry)

FONT LIST
   default: any device-dependent font that can represent the default character set

all BUNDLE INDEXES
   default: 1

all ASPECT SOURCE FLAGS
   default: individual

LINE TYPE
   default: solid

LINE WIDTH
   default: 1/1000 of the maximum length of the longest side of the rectangle defined by VDC extent for 'absolute'; 1.0 for 'scaled'

LINE COLOR
   default: 1 for 'indexed'; device-dependent foreground color for 'direct'

MARKER TYPE
   default: asterisk

MARKER SIZE
   default: 1/100 of the maximum length of the longest side of the rectangle defined by VDC extent for 'absolute'; 1.0 for 'scaled'

MARKER COLOR
   default: 1 for 'indexed'; device-dependent foreground color for 'direct'

INTERIOR STYLE
   default: hollow, perimeter visibility 'on'

FILL COLOR
   default: 1 for 'indexed'; device-dependent foreground color for 'direct'

HATCH INDEX
   default: 1

PATTERN INDEX
   default: 1

PATTERN TABLE
   default: 1; m=n=1; device dependent foreground color

PATTERN SIZE
   default: 1/25 of the maximum length of the longest side of the rectangle defined by VDC extent

PATTERN REFERENCE POINT
   default: 0,0

PERIMETER COLOR
   default: 1 for 'indexed'; device-dependent foreground color for 'direct'

PERIMETER TYPE
   default: solid

PERIMETER WIDTH
   default: 1/1000 of the maximum length of the longest side of the rectangle defined by VDC extent for 'absolute'; 1.0 for 'scaled'

TEXT FONT INDEX
   default: 1

TEXT PRECISION
   default: string

CHARACTER EXPANSION FACTOR
   default: 1

CHARACTER SPACING
   default: 0

TEXT COLOR
   default: 1 for 'indexed'; device-dependent foreground color for 'direct'

CHARACTER HEIGHT
   default: 1/100 of the maximum length of the longest side of the rectangle defined by VDC extent

CHARACTER ORIENTATION
   default: (0,1,1,0)

CHARACTER PATH
   default: right

TEXT ALIGNMENT
   default: normal horizontal, normal vertical

COLOR SELECTION MODE
   default: indexed

COLOR TABLE
   default: device-dependent background color for index = 0; device-dependent foreground color for index = 1

LOCAL BACKGROUND COLOR
default: on-off switch = off; 0 for 'indexed'; device-dependent background color for 'direct'

LINE WIDTH SPECIFICATION MODE
default: scaled

MARKER SIZE SPECIFICATION MODE
default: scaled

PERIMETER WIDTH SPECIFICATION MODE
default: scaled

APPENDIX A

FORMAL SPECIFICATION

This Appendix is not a part of American National Standard X3.xxx-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

Section A.1 contains a formal grammar, and Section A.2 contains a state diagram and related information.

A.1 Formal Grammar

This grammar is a formal definition of a standard VDM syntax. The encoding-independent and the encoding-dependent productions are separated, and there are two subsections showing the syntax of the two standardized encoding schemes. Details on the encoding of terminal symbols can be found in sections of this standard that deal with the particular encoding schemes.

Notation used:

<symbol>	- nonterminal
<SYMBOL>	- terminal
<symbol>*	- 0 or more occurrences
<symbol>+	- 1 or more occurrences
<symbol>o	- optional, 0 or 1 occurrence
<symbol>(n)	- exactly n occurrences; n=2,3,...
<symbol-1> ::= <symbol-2>	- symbol-1 has the syntax of symbol-2
<symbol-1> | <symbol-2>	- symbol-1 or alternatively symbol-2
<symbol: meaning>	- symbol with the stated meaning
(comment)	- explanation of a symbol or a production

A.1.1 Metafile Elements

```
<metafile>                  ::= <BEGIN METAFILE>
                                  <character substitutions>o
                                  <identifier>
                                <metafile descriptor>
                                <metafile contents>*
                                <END METAFILE>
```

```
<metafile contents>          ::= <picture>
                               |  <external element>

<picture>                    ::= <BEGIN PICTURE>
                                   <identifier>
                                   <on-off flag enumerated>
                                   <color>
                                 <picture descriptor element>*
                                 <BEGIN PICTURE BODY>
                                 <picture element>*
                                 <END PICTURE>

<on-off flag enumerated>     ::= <on>
                               | <off>
```

A.1.2 METAFILE DESCRIPTOR ELEMENTS

```
<metafile descriptor>        ::= <identification>
                                 <characteristics>

<identification>             ::= <VDM VERSION>
                                   <integer>
                                 <vdm description>o

<vdm description>            ::= <VDM DESCRIPTION>
                                   <string>

<characteristics>            ::= <VDM ELEMENT LIST>
                                   <element name enumerated>*
                                 <optional descr elmt>*

<optional descr elmt>        ::= <VDC TYPE>
                                   <vdc type>
                               | <MAXIMUM COLOR INDEX>
                                   <color index>
                               | <VDM DEFAULTS REPLACEMENTS>
                                   <element default>*
                               | <FONT LIST>
                                   <font name>*
                               | <CHARACTER SET LIST>
                                   <character set definition>*
                               | <scalar precision>*
                               | <external element>

<vdc type>                   ::= <INTEGER>
                               | <REAL>

<element default>            ::= <control element>
                               | <attribute element>

<font name>                  ::= <string>
```



```
<character set definition>   ::= <char set enumerated>
                                   <designation sequence>

<index>                      ::= <standard index value>
                               | <private index value>

<standard index value>       ::= <non-negative integer>
<non-negative integer>       ::= <integer>(greater or equal to 0)
<positive integer>           ::= <integer>     (greater than 0)
<private index value>        ::= <negative integer>
<negative integer>           ::= <integer>     (less than 0)
<positive index>             ::= <positive integer>

<char set enumerated>        ::= <94 CHAR>
                               | <96 CHAR>
                               | <MULTI-BYTE 94 CHAR>
                               | <MULTI-BYTE 96 CHAR>
                               | <COMPLETE CODE>

<designation sequence>       ::= <string>

<scalar precision>           ::= <NON-VDC INTEGER PRECISION>
                                   <integer prec value>
                               | <NON-VDC REAL PRECISION>
                                   <real prec value>
                               | <INDEX PRECISION>
                                   <index prec value>
                               | <COLOR PRECISION>
                                   <color prec value>
                               | <COLOR INDEX PRECISION>
                                   <col index prec value>
                                 (these elements have encoding)
                                 (dependent parameters        )
```

A.1.3 PICTURE DESCRIPTOR ELEMENTS

```
<picture descriptor element> ::= <COLOR SELECTION MODE>
                                   <color select mode>
                               | <SCALING MODE>
                                   <scaling spec mode>
                                   <metric scale factor>
                               | <LINE WIDTH SPECIFICATION MODE>
                                   <spec mode>
                               | <MARKER SIZE SPECIFICATION MODE>
                                   <spec mode>
                               | <PERIMETER WIDTH SPECIFICATION MODE>
                                   <spec mode>
                               | <external element>

<color select mode>          ::= <INDEXED>
                               | <DIRECT>
```

```
<scaling spec mode>          ::= <ABSTRACT>
                               | <METRIC>

<metric scale factor>        ::= <real>

<spec mode>                  ::= <ABSOLUTE>
                               | <SCALED>
```

A.1.4 PICTURE ELEMENTS

```
<picture element>            ::= <control element>
                               | <graphical element>
                               | <attribute element>
                               | <escape element>
                               | <external element>
```

A.1.5 CONTROL ELEMENTS

```
<control element>            ::= <BACKGROUND COLOR>
                                   <on-off flag enumerated>
                                   <color>
                               | <VDC EXTENT>
                                   <coordinate>(2)
                               | <CLIP RECTANGLE>
                                   <vdc value>(4)
                               | <CLIP INDICATOR>
                                   <on-off flag enumerated>
                               | <coordinate precision>

<coordinate precision>       ::= <INTEGER VDC PRECISION>
                                   <integer vdc prec value>
                               | <REAL VDC PRECISION>
                                   <real vdc prec value>
                                  {these elements have encoding}
                                  {dependent parameters        }
```

A.1.6 GRAPHICAL ELEMENTS

```
<graphical element>          ::= <polypoint element>
                               | <circle element>
                               | <arc element>
                               | <text element>
                               | <cell element>
```



```
<polypoint element>          ::= <POLYLINE>
                                   <coordinate>(1)
                                   <coordinate list>
                               | <POLYMARKER>
                                   <coordinate>
                                   <coordinate list>
                               | <POLYGON>
                                   <coordinate>(3)
                                   <coordinate list>

<coordinate list>            ::= <coordinate>*

<circle element>             ::= <CIRCLE>
                                   <coordinate>
                                   <radius>

<radius>                     ::= <vdc value>

<arc element>                ::= <ARC>
                                   <coordinate>(3)
                               | <ARC CLOSE>
                                   <coordinate>(3)
                                   <close type>

<close type>                 ::= <PIE>
                               | <CHORD>

<text element>               ::= <TEXT>
                                   <coordinate>
                                   <text tail>

<text tail>                  ::= <final character list>
                               | <nonfinal character list>

<final character list>       ::= <FINAL>
                                   <character list>

<nonfinal character list>    ::= <NOT FINAL>
                                   <string>
                                   <character attribute
                                    element>*
                                   <spanned text>

<spanned text>               ::= <APPEND TEXT>
                                   <text tail>

<cell element>               ::= <CELL ARRAY>
                                   <coordinate>(3)
                                   <integer>(2)
                                   <color list>
```

A.1.7 ATTRIBUTE ELEMENTS

```
<attribute element>          ::= <aspect source flags>
                               | <line attribute element>
                               | <marker attribute element>
                               | <area attribute element>
                               | <text attribute element>
                               | <color table element>
<aspect source flags>        ::= <SET ASPECT SOURCE FLAGS>
                                   <asf pair>+

<asf pair>                   ::= <asf number>
                                 <asf>

<asf number>                 ::= <integer>     (0 to 17)

<asf>                        ::= <INDIVIDUAL>
                               | <BUNDLED>

<line attribute element>     ::= <POLYLINE BUNDLE INDEX>
                                   <positive index>
                               | <LINE TYPE>
                                   <index>
                               | <LINE WIDTH>
                                   <scale value>
                               | <LINE COLOR>
                                   <color>

<scale value>                ::= <vdc value>
                               | <real>

<color>                      ::= <color index>
                               | <red green blue>

<marker attribute element>   ::= <POLYMARKER BUNDLE INDEX>
                                   <positive index>
                               | <MARKER TYPE>
                                   <index>
                               | <MARKER SIZE>
                                   <scale value>
                               | <MARKER COLOR>
                                   <color>
```



```
<area attribute element>     ::= <FILL AREA BUNDLE INDEX>
                                   <positive index>
                               | <INTERIOR STYLE>
                                   <interior enumerated>
                                   <on-off flag enumerated>
                               | <FILL COLOR>
                                   <color>
                               | <HATCH INDEX>
                                   <positive index>
                               | <PATTERN INDEX>
                                   <positive index>
                               | <PATTERN REFERENCE POINT>
                                   <coordinate>
                               | <PATTERN TABLE>
                                   <positive index>
                                   <integer>(2)
                                   <color> (integer * x
                                            integer 2)
                               | <PATTERN SIZE>
                                   <vdc value>(2)
                               | <PERIMETER TYPE>
                                   <index>
                               | <PERIMETER WIDTH>
                                   <scale value>
                               | <PERIMETER COLOR>
                                   <color>

<interior enumerated>        ::= <HOLLOW>
                               | <SOLID>
                               | <HATCH>
                               | <PATTERN>

<text attribute element>     ::= <char attribute element>
                               | <string attribute element>

<char attribute element>     ::= <TEXT BUNDLE INDEX>
                                   <positive index>
                               | <TEXT FONT INDEX>
                                   <positive index>
                               | <TEXT PRECISION>
                                   <text precision enumerated>
                               | <CHARACTER EXPANSION FACTOR>
                                   <real>
                               | <CHARACTER SPACING>
                                   <real>
                               | <TEXT COLOR>
                                   <color>
                               | <CHARACTER HEIGHT>
                                   <vdc value>
                               | <CHARACTER SET INDEX>
                                   <positive index>
```

```
<text precision enumerated>  ::= <STRING>
                               | <CHARACTER>
                               | <STROKE>

<string attribute element>   ::= <CHARACTER ORIENTATION>
                                   <vdc value>(4)
                               | <CHARACTER PATH>
                                   <path enumerated>
                               | <TEXT ALIGNMENT>
                                   <horizontal align enumerated>
                                   <vertical align enumerated>
                                   <continuous align value>o

<continuous align value>     ::= <real>(2)

<path enumerated>            ::= <RIGHT>
                               | <LEFT>
                               | <UP>
                               | <DOWN>

<horizontal align enumerated> ::= <LEFT>
                               | <CENTER>
                               | <RIGHT>
                               | <NORMAL HORIZONTAL>
                               | <CONTINUOUS HORIZONTAL>

<vertical align enumerated>  ::= <TOP>
                               | <CAP>
                               | <HALF>
                               | <BASE>
                               | <BOTTOM>
                               | <NORMAL VERTICAL>
                               | <CONTINUOUS VERTICAL>
```

A.1.8 ESCAPE ELEMENT

```
<escape element>             ::= <VDM ESCAPE>
                                   <integer>
                                   <escape data list>
```

A.1.9 EXTERNAL ELEMENTS

```
<external element>           ::= <MESSAGE>
                                   <action flag>
                                   <string>
                               | <APPLICATION DATA>
                                   <string>
                                   <string>

<action flag>                ::= <TRUE>
                               | <FALSE>
```


A.1.10 TERMINAL SYMBOLS

The following are the terminals in this grammar. Their representation is dependent on the encoding scheme used. In Appendix A of the subsequent parts of this standard, these encoding-dependent symbols are further described.

```
<coordinate>
<integer>
<real>
<vdc value>
<string>
<character substitution>
<color index>
<red green blue>
<integer prec value>
<real prec value>
<index prec value>
<color prec value>
<col index prec value>
<integer vdc prec value>
<real vdc prec value>
<color list>
<escape data list>
```

The VDM opcodes are encoding dependent. A complete list of them can be found in the productions for <element name enumerated> below.

The enumerated types:

```
<INTEGER>     <REAL>
<ON>          <OFF>
<INDEXED>     <DIRECT>
<ABSTRACT>    <METRIC>
<ABSOLUTE>    <SCALED>
<94 CHAR>     <96 CHAR>   <MULTI-BYTE 94 CHAR>
<MULTI-BYTE 96 CHAR>      <GRAPHICS CODING SYSTEMS>
<PIE>         <CHORD>
<FINAL>       <NOT FINAL>
<INDIVIDUAL>  <BUNDLED>
<HOLLOW>      <SOLID>     <HATCH>     <PATTERN>
<STRING>      <CHARACTER> <STROKE>
<RIGHT>       <LEFT>      <UP>        <DOWN>
<CENTER>      <NORMAL HORIZONTAL>     <CONTINUOUS HORIZONTAL>
<TOP>         <CAP>       <HALF>      <BASE>       <BOTTOM>
<NORMAL VERTICAL>  <CONTINUOUS VERTICAL>  <YES>  <NO>

<element name enumerated>    ::= <BEGIN METAFILE>
                                | <END METAFILE>
```

| <VDM DESCRIPTION>
| <VDM VERSION>
| <VDC TYPE>
| <INTEGER PRECISION>
| <REAL PRECISION>
| <INDEX PRECISION>
| <COLOR PRECISION>
| <COLOR INDEX PRECISION>
| <MAXIMUM COLOR INDEX>
| <VDM ELEMENT LIST>
| <VDM DEFAULTS REPLACEMENT>
| <FONT LIST>
| <CHARACTER SET LIST>
| <BEGIN PICTURE>
| <BEGIN PICTURE BODY>
| <END PICTURE>
| <LOCAL BACKGROUND COLOR>
| <INTEGER VDC PRECISION>
| <REAL VDC PRECISION>
| <VDC EXTENT>
| <CLIP RECTANGLE>
| <CLIP INDICATOR>
| <SCALING MODE>
| <COLOR SELECTION MODE>
| <LINE WIDTH SPECIFICATION MODE>
| <MARKER SIZE SPECIFICATION MODE>
| <PERIMETER WIDTH SPECIFICATION MODE>
| <POLYLINE>
| <POLYMARKER>
| <POLYGON>
| <CIRCLE>
| <ARC>
| <ARC CLOSE>
| <TEXT>
| <APPEND TEXT>
| <CELL ARRAY>
| <SET ASPECT SOURCE FLAGS>
| <POLYLINE BUNDLE INDEX>
| <LINE TYPE>
| <LINE WIDTH>
| <LINE COLOR>
| <POLYMARKER BUNDLE INDEX>
| <MARKER TYPE>
| <MARKER SIZE>
| <MARKER COLOR>
| <FILL AREA BUNDLE INDEX>
| <INTERIOR STYLE>
| <FILL COLOR>
| <HATCH INDEX>
| <PATTERN INDEX>
| <PATTERN TABLE>
| <PATTERN REFERENCE POINT>

| <PATTERN SIZE>
| <PERIMETER TYPE>
| <PERIMETER WIDTH>
| <PERIMETER COLOR>
| <CHARACTER SET INDEX>
| <TEXT BUNDLE INDEX>
| <TEXT FONT INDEX>
| <TEXT PRECISION>
| <CHARACTER EXPANSION FACTOR>
| <CHARACTER SPACING>
| <TEXT COLOR>
| <CHARACTER HEIGHT>
| <CHARACTER ORIENTATION>
| <CHARACTER PATH>
| <TEXT ALIGNMENT>
| <COLOR TABLE>
| <VDM ESCAPE>
| <MESSAGE>
| <APPLICATION DATA>

A.2 VDM State Diagram

The preceding sections of this appendix give the formal grammar of the VDM. Embedded in the formal grammar are some required sequential relationships between elements, relationships that must pertain in a syntactically legal metafile. For example, the Metafile Descriptor (which is the first sequence of consecutive elements classified as Metafile Descriptor elements) must occur in a metafile after the BEGIN METAFILE element and before any other elements (disregarding external elements).

While the formal grammar is complete and precise, there are simpler ways of illustrating the required rules of sequentiality. One such method is a state diagram. Because a metafile is a static data structure and not a process, it doesn't really have states. Nevertheless, the state notion is useful when applied to metafiles.

Consider an abstract machine that can traverse the metafile data structure from beginning to end and is able to identify or comprehend (as opposed to interpret, render, or display) the metafile elements, and has a single significant structural component called a *state register*. The identification by this abstract machine of various metafile elements in the sequential data structure causes the state register to assume certain values. The metafile "states" in the following state diagram are then actually the values of the state register of this abstract metafile-traversing machine.

While it may seem that this presentation of a state diagram is a standardization of metafile interpreters, it is in fact not so. The interpretation of a metafile has two components. The first is the identification of the individual metafile elements. This identification is solely related to the parsing and comprehension of metafile syntax. Because the syntax is standardized and conformance rules for it are given, the identification component of any metafile interpreter is well-defined. The second component of metafile interpretation is the interpretation, rendering, and display of the elements comprising the virtual picture definition (that is, the presentation of a physical picture based on the virtual picture definition). Conformance requirements for this component of metafile interpretation are explicitly excluded from this VDM standard (although guidelines are given in a later appendix for those interested in achieving uniformity of results).

Table A-1 lists those elements that cause transitions from one state to another state. Figure A-1 and the table contain both names of individual VDM elements (for example, BEGIN PICTURE) names referring to groups of elements (for example, PICTURE ELEMENT). The individual VDM elements comprising these latter groups can be found by referring to the formal grammar in the preceding sections of this appendix. Note that for the purposes

of this diagram and table, escape elements may occur anywhere that external elements may occur.

Table A-1: VDM Elements and States

Element/Starting State	Resulting State
BEGIN METAFILE/Initial	Metafile Description
METAFILE DESCRIPTOR/Metafile Description	Metafile Description
BEGIN PICTURE/Metafile Description	Picture Description
BEGIN PICTURE/Picture Closed	Picture Description
PICTURE DESCRIPTOR/Picture Description	Picture Description
BEGIN PICTURE BODY/Picture Description	Picture Open
PICTURE ELEMENT/Picture Open	Picture Open
END PICTURE/Picture Description	Picture Closed
END PICTURE/Picture Open	Picture Closed
END METAFILE/Metafile Description	Final
END METAFILE/Picture Closed	Final
EXTERNAL ELEMENT/Any State	Same State

NOTE: Any other element/state combinations are illegal and can be considered to cause transition to an error state. For the purposes of this diagram escape elements may occur anywhere that external elements may occur.

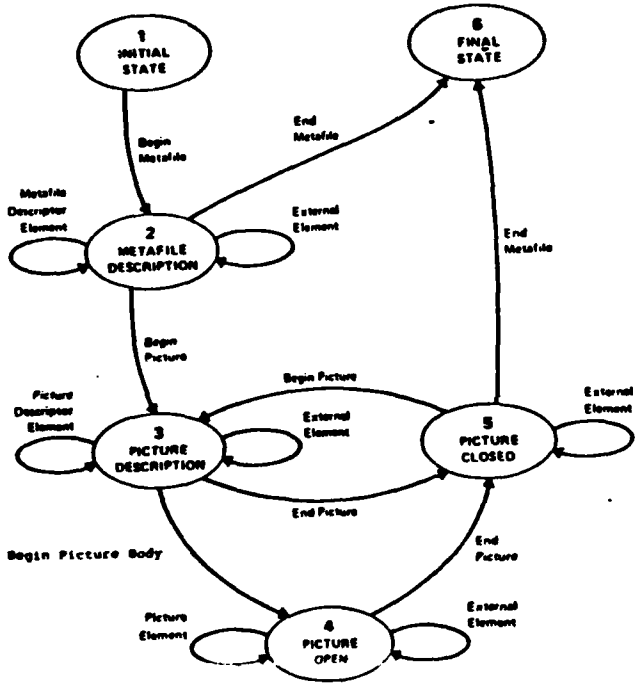


Figure A-1: VDM state diagram.

APPENDIX B

DESIGN PRINCIPLES

This Appendix is not a part of American National Standard X3.xxx-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

Following these principles will help to ensure well-designed, implementable standards and will help to resolve issues during the development process. Conflicts among design principles will be resolved on an issue-by-issue basis. Definitions of principles and concepts will be consistent with the WG2 recommendations (ISO/TC97/SC5/WG2 N54) and the X3H3 working vocabulary (X3H3/81-23).

- Completeness. Functions within any area of the standard will be included in their entirety.
- Conciseness. Redundant elements or parameters will be avoided.
- Consistency. Contradictory elements will be avoided.
- Extensibility. The ability to add new elements and generality to the standard will not be precluded.
- Fidelity. The minimal results and characteristics of elements will be well defined.
- Implementability. An element will be able to be efficiently supported on most host systems and/or graphics hardware.
- Orthogonality. Independent elements for separate and noninteracting activities will be provided. For example, text attributes should not cause interaction with the POLYLINE element.
- Predictability. The standards will be such that the recommended or proper use of standard elements will guarantee what the results of using a particular element will be.
- Standard practice. Only those elements that reflect existing practice, that are necessary to support existing practice, or that are necessary to support proposed standards will be standardized.
- Usefulness. Functions will be powerful enough to perform useful tasks.

- well-structured. The assumptions that elements make about each other will be minimized. An element will have a well-defined interface and a simply stated unconditional purpose. Multipurpose elements and side effects will be avoided.

APPENDIX C

REFERENCE MODELS

This Appendix is not a part of American National Standard X3.xxx-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

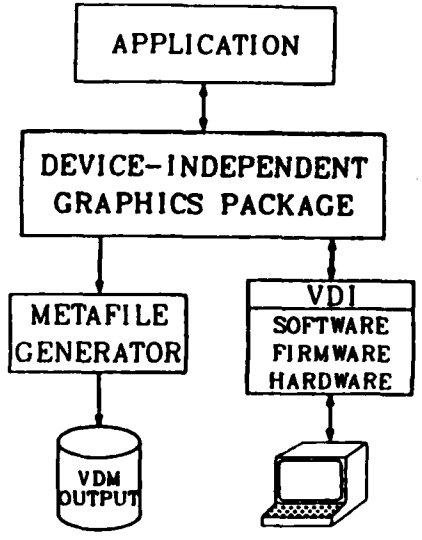


Figure C-1: Relationship of VDM to traditional graphics package.

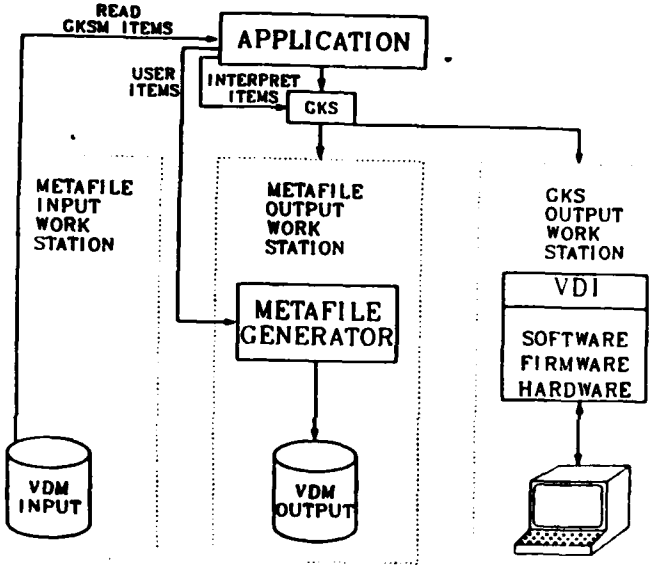


Figure C-2: Relationship of VDM to GKS.

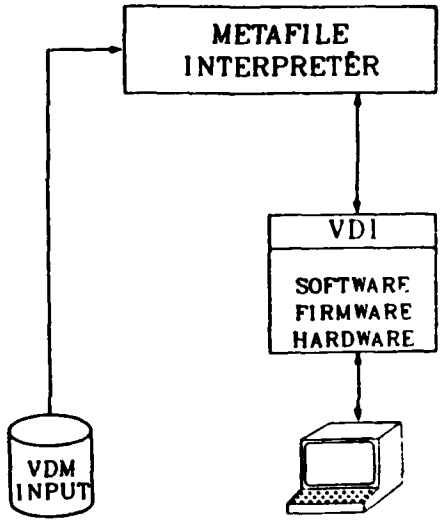


Figure C-3: Metafile interpretation with no graphics package.

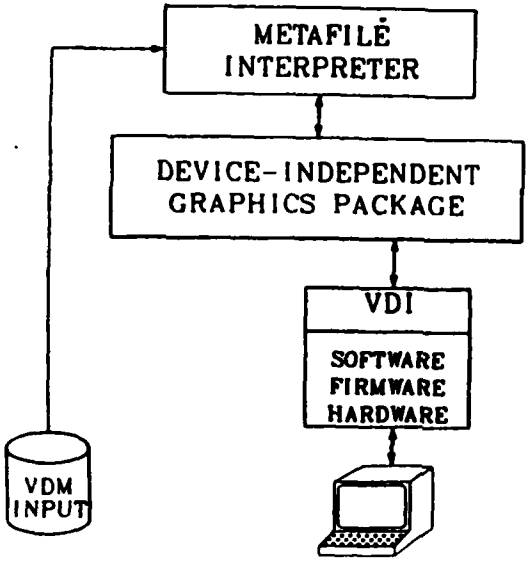


Figure C-4: Metafile interpretation using traditional graphics package.

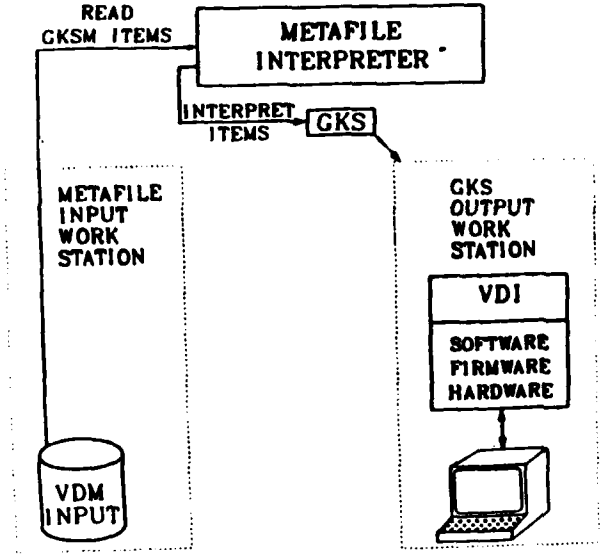


Figure C-5: Metafile interpretation using GKS.

APPENDIX D

VDM INTERPRETER GUIDELINES

This Appendix is not a part of American National Standard X3.xxx-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

The VDM standardizes the contents, syntax, and semantics of a set of VDM elements. It does not standardize the metafile interpreter. Nonetheless, for predictability and uniformity of results, it is useful to suggest a common approach to situations where a metafile interpreter cannot accurately render the contents of a metafile.

D.1 Introduction

Sections D.2 through D.5 contain recommended approximations for the interpretation of VDM elements where no one-to-one mapping exists between a VDM element and display device capability. Section D.6 lists minimum required capabilities for VDM interpreters.

D.2 General

An out-of-range index in an indexed selection element causes selection of the default index value; an out-of-range index on an index definition element (for example, COLOR TABLE) is ignored.

D.3 Viewing

If CLIP RECTANGLE is outside VDC EXTENT, clipping should occur at the intersection of CLIP RECTANGLE and VDC EXTENT.

CLIPPING OF GRAPHICAL ELEMENTS: If a graphical element is clipped by clipping specified in the metafile, the clip boundary is not considered part of the element and it is recommended that this boundary not be drawn. This applies to POLYLINE, POLYGON, ARC CLOSE, and CIRCLE.

ASPECT SOURCE FLAGS: If the initial ASFs are not altered, the expected behavior of the interpreter is

(a) as if individual specification of bundled aspects were not a system feature, if the initial values of all the ASFs are 'bundled'; or

(b) as if specification of bundled aspects via a bundle were not a system feature, if the initial values of all the ASFs are 'individual'.

Polygon: Sufficient storage to permit polygons containing up to 128 vertices should be provided.

Text: Sufficient storage to permit accumulation and alignment of 80 characters plus a reasonable number of attribute changes should be provided.

D.4 Control

It is intended that the BEGIN PICTURE element clears the view surface. This is necessary if pictures are randomly accessed. If the metafile reader is composing an image from multiple VDM pictures, the interpreter will clear the view surface only before the first picture in the image.

BEGIN PICTURE: The element typically causes the view surface to be cleared to the specified background color.

END PICTURE: It is suggested that interpretation of the END PICTURE element guarantee that the displayed picture reflect the execution of all the elements in the picture body.

LOCAL BACKGROUND COLOR: The interpreter must always ensure distinguishability of bundle items if all asf's are set to bundle. For example, for a device that must have a local background color, bundled polylines cannot be distinguished by line type if both LOCAL BACKGROUND COLOR and LINE COLOR are the same value.

D.5 Attributes

BUNDLES: Section 4 describes the component attributes comprising the various attribute bundles. To avoid possible conflict with future revisions of the standard, it is strongly recommended that interpreters use only these components to achieve distinguishability and not use other, currently nonstandardized attributes. If, however, it is not feasible for the interpreter to adhere to this guideline, then the use of device-dependent, nonstandardized attributes (for example, blink or highlight) is a reasonable alternative.

LINE TYPE: If the specified implementation-dependent line type is not available, 'solid' is used.

LINE WIDTH: An absolute line width value of 0 is interpreted as the narrowest line width of the device. If a device cannot produce a line of the exact specified width, the closest

implemented width is chosen.

MARKER TYPE: If the specified implementation-dependent marker type is not available, the marker type 'asterisk' is used.

MARKER SIZE: The marker size is mapped to the nearest available marker size on the device. The effect of MARKER SIZE on implementation-dependent markers is implementation dependent. An absolute marker size value of 0 is interpreted as the smallest distinguishable marker.

INTERIOR STYLE: If the requested interior style is not available, 'hollow' is used.

PATTERN SIZE: If a device cannot produce a pattern of the exact specified size, the closest implemented size is chosen.

TEXT PRECISION: If a specified text precision is not available, the next more precise implemented text precision is chosen. If no such (more precise) value is available, it is interpreter dependent whether: (1) the next lower value is used, or (2) whether a substitute font will be involved temporarily to provide the requested text precision. NOTE: the font may already be a temporary invocation to provide a character set not provided by the current TEXT FONT INDEX (see Appendix D, CHARACTER SET INDEX).

CHARACTER EXPANSION FACTOR: The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected. The effective character height and character width are set to values that allow characters to completely fit into an enclosing rectangle determined by the desired character height and width. If this is not possible, the smallest available character size is used.

CHARACTER SPACING: The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected.

CHARACTER HEIGHT: The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected.

CHARACTER ORIENTATION: If the specified character up vector is not available, the nearest available vector is chosen. If two are equally near, the one in a counterclockwise direction is chosen. When the character up vector and the character base vector are not at right angles, and hardware text is being used which cannot be skewed, the base vector is used to determine character orientation. If the character path is 'left' or 'right', the character base vector determines placement of character origins; if the path is 'up' or 'down', the character

up vector determines placement of character origin.

CHARACTER PATH: The fallback value for 'left' is 'right', the fallback value for 'up' is 'down', and vice versa. If the fallback value recommended above is not available, 'right' is chosen.

TEXT ALIGNMENT: Horizontal alignment is 'left' if 'center', 'right', or 'continuous horizontal' are not available. Vertical alignment is 'base' if a specified value is not available.

CHARACTER SET AND TEXT FONT: If the currently selected character set cannot be rendered in the current font, it is suggested that the font be temporarily overridden and a font be used which can represent the character set (that is, the character set should be considered to have precedence over the font).

FONT DESIGN AND THE FONT COORDINATE SYSTEM: Sufficient white space should be allowed when setting the limits of the character body relative to the characters in the font to permit characters to be displayed with their bodies flush without producing conflicts or overlaps between ascenders and descenders, and with normal spacing between characters. This permits standardized use of the continuous alignment parameter in TEXT alignment, and supports the intent of spacing with CHARACTER SPACING = 0.0.

CHARACTER SET INDEX: If the character set selected is not available in the selected font at the time of elaboration of TEXT or APPEND TEXT elements, the font will be temporarily changed to one where the selected character set can be represented.

8.6 Graphical Elements

CELL ARRAY: A device that cannot display the CELL ARRAY element draws a rectangle corresponding to the specified rectangular area. The rectangle is drawn according to the fill area perimeter attributes without regard to the perimeter visibility parameter.

If dx or dy is zero in the cell array specification or if the parallelogram definition leads to a degenerate (zero area) parallelogram, it is suggested that no output be generated by the metafile interpreter.

If the three points defining the CELL ARRAY form a parallelogram and the CELL ARRAY cannot be displayed as a parallelogram, then it is implementation dependent whether the specified parallelogram is drawn or the CELL ARRAY is displayed without skewing.

POLYGON: A polygon is geometrically degenerate if its vertex points are collinear or result in only one distinct vertex. A line is drawn through collinear vertices, and a dot is drawn for a

single distinct vertex. This line or dot is subject to the interior attributes if perimeter visibility is 'off' and is subject to the perimeter attributes if perimeter visibility is 'on'.

CIRCLE: For a circle with zero radius, a dot is displayed. A negative radius is treated as zero. This dot is subject to the interior attributes if perimeter visibility is 'off', and is subject to the perimeter attributes if perimeter visibility is 'on'.

ARC: A dot is drawn for an ARC with only one distinct point. An ARC element with only two distinct coordinates is ignored. It is not recommended that a circle is drawn whose diameter is the line between the two distinct points because the size and position of that circle bears little relationship to an arc which becomes degenerate because of accumulated computation error. If an ARC element has three colinear coordinates, the arc has zero curvature and a line is drawn through the three coordinates.

ARC CLOSE: A dot is drawn for an ARC CLOSE with only one distinct point. An ARC CLOSE element with only two distinct points is ignored for reasons stated above for ARC. If a 'chord' closed arc has three colinear coordinates, a line is drawn through the three coordinates. A 'pie' closed arc with three colinear coordinates is ignored because it does not have a finite or unique center point. If a dot or line is drawn, this dot or line is subject to the perimeter attributes if perimeter visibility is 'on' and to the interior attributes if perimeter visibility is 'off'.

8.7 Color Model

COLOR MODEL: There are two key assumptions to the formation of direct and indexed color specifications:

(1) Many computer graphics devices are inherently indexed in regard to color attribute selection. These devices include the following:

(a) frame buffer video bit maps (index is pixel integer);

(b) pen plotters (index is pen id);

(c) stroke devices (index is analog of voltage for refresh intensity or beam penetration).

(2) All of these devices map these indexes to visible attributes through a map. These maps are either loadable under program control (type a above), operator control (b), or are fixed in hardware (or an e may be accessible in software at some level.

Thus, direct hardware control over these devices is, thus, by color selection by index. Direct hardware control (by index control) presumes that software must have enough knowledge of the accessible hardware range to be able to control the device usefully.

To utilize direct color specification (if, by assumption 1, devices implement indexed specifiers), we need to reverse-map the ones noted in assumption 2.

A "closest match" element between the directly specified color and the entries of the table could be based on several algorithms. For example:

(1) Minimum spatial distance, as computed in the RGB color cube.

(2) Component by-component comparison by minimum weight NORing. In fixed tables, colors tend to be evenly distributed throughout the color space, or at least across a plane through the space. This aids the closest match problem because there is likely to be a color somewhat near the target.

In loadable tables of indeterminate length, it may be advisable to load colors from disparate points in the color space early in the table before the color selection table is exhausted.

Only the COLOR TABLE element can change the color map (if it exists). Direct specification requires that closest match be used to protect static attributes in a display, which, for portability, must be the default.

In *American color television systems (NTSC encoding)*, color signals are translated to gray scale for monochrome reception by the following equation:

$$Y = 0.30*R + 0.59*G + 0.11*B$$

where R, G, and B are the intensity values of the red, green, and blue components, and Y is the resulting luminance value.

The integer mapping of this equation would be, approximately

$$Y' = 3*R + 6*G + 1*B$$

and a simpler system would be

$$y'' = B + 2*R + 4*G$$

which preserves binary orders of magnitude.

Another algorithm for mapping color specification to monochrome intensity, and the one mandated by the GKS standard for monochrome workstations, is

$$\text{intensity} = 0.5 * (M + m)$$

where

$$M = \max(\text{red}, \text{green}, \text{blue}) \quad m = \min(\text{red}, \text{green}, \text{blue})$$

The computed intensity is then mapped to the nearest available on the workstation.

B.8 Minimum Required Capability List

For uniformity of results when interpreting a metafile, it is suggested that VDM interpreters have at least the following capabilities.

Capability	Minimum Required Interpreter Support
POLYLINE BUNDLE INDEX	5
LINE TYPE	solid,dash,dot,dash-dot, dash-dot-dot
LINE WIDTH	1, interpreter dependent
LINE COLOR	1, interpreter dependent
POLYMARKER BUNDLE INDEX	5
MARKER TYPE	dot,plus,asterisk,circle,cross
MARKER SIZE	1, interpreter dependent
MARKER COLOR	1, interpreter dependent
FILL AREA BUNDLE INDEX	5
INTERIOR STYLE	hollow,solid,hatch,pattern
FILL COLOR	1, interpreter dependent
HATCH INDEX	1, interpreter dependent
PATTERN INDEX	1, interpreter dependent
PATTERN SIZE	1, interpreter dependent
TEXT BUNDLE INDEX	2
CHARACTER SET INDEX	2
TEXT FONT INDEX	1
TEXT PRECISION	string,character
CHARACTER EXPANSION FACTOR	1, interpreter device dependent
CHARACTER SPACING	1, interpreter dependent
TEXT COLOR	1, interpreter dependent
CHARACTER HEIGHT	1, interpreter dependent
CHARACTER UP VECTOR	along the y-axis of VDC space
CHARACTER PATH	right,left,up,down
TEXT ALIGNMENT	top, bottom, left, right, center, baseline
LOCAL BACKGROUND COLOR	1, interpreter dependent
PERIMETER COLOR	Same as LINE COLOR
PERIMETER TYPE	Same as LINE TYPE
PERIMETER WIDTH	Same as LINE WIDTH
CHARACTER SET LIST	ASCII
TEXT FONT LIST	at least one font capable of displaying the ASCII character set
TEXT string size for alignment	80 characters
Polygon vertices for area fill	128

PART 2. CHARACTER CODED GRAPHICS BINDING FOR THE VIRTUAL DEVICE METAFILE

1. Introduction

This standard specifies functionality separately from coding formats. This allows the same functionality to be bound to several encoding schemes. One such coding scheme, character coded graphics, which is based on the code extension techniques of ISO International Standard 2022-1982 is included here.

1.1 Criteria for Character Coded Graphics Binding. This coding scheme is intended to be used in situations in which it is important to minimize the number of characters that are recorded in a file or transmitted over data communications lines, even though some extra processing may be required. For instance, when the metafile is transmitted over relatively slow-speed serial data communications lines (for example, 9600 bps or less), speedy presentation of the data requires that the number of characters transmitted be kept to a minimum.

If minimizing the processing overhead is more important than data compaction, a coding scheme other than this one may be a better choice. For example, the binding contained in Part 3 of this standard.

Each VDM command follows a simple regular syntax. Thus, new commands can be added in a future revision of the standard such that existing VDM interperters can recognize (and ignore) the new commands. Also, new operands can be added to existing commands in the future revision of the standard such that existing VDM interperters can recognize (and ignore) the additional operands.

Each VDM operand follows a simple regular syntax which is variable length. This permits small values to be represented by one or a small number of bytes.

A certain range of operand values of standard commands have been reserved for private use; the remaining range is either standardized or reserved for future standardization. This reserving for private use permits "value-added" implementations in a controlled way that will never be allocated standard meaning in future revisions of the standard.

1.2 Conformance. A metafile conforms to this character-coded binding if it meets the following requirements:

- Each metafile element described in this section is coded in the manner described.
- Private (nonstandard) metafile elements are all coded using the VDM ESCAPE metafile element. Opcodes reserved for

future standardization are not used to code private (nonstandard) metafile elements.

- Private (nonstandard) values of index parameters are all coded using negative integers. In coding index parameters, a metafile shall not use nonnegative integers to represent private values of index parameters.

A conforming metafile may include, within the string parameters of TEXT and APPEND TEXT metafile elements, the ISO 2022 controls for designating and invoking G-sets. This is an alternative way, in addition to CHARACTER SET INDEX, by which character sets for displaying text strings may be selected. However, the use of ISO 2022 controls within text strings is implementation dependent. Metafile interpreters are not required to respond correctly to the ISO 2022 controls for designating and invoking G-sets when those controls occur within a TEXT or APPEND TEXT command.

2. Notational Conventions

2.1 7-Bit and 8-Bit Code Tables. In a 7-bit environment, the code table ("in-use table") is represented as a table of eight columns and sixteen rows. A byte, or bit combination, has seven bits, numbered b1 to b7, from least significant to most significant. Bits b7, b6, and b5 address the columns, and bits b4, b3, b2, and b1 address the rows. A "col/row" notation is used in which "col" is the decimal column number, and "row" is the decimal row number. For example, "1/11" refers to the bit combination in column 1, row 11 of the code chart: binary 0011011. Figure 1 shows the 7-bit code table.

In an 8-bit environment, the code table has sixteen columns and sixteen rows. Bits b8, b7, b6, and b5 address the columns, and bits b4, b3, b2, and b1 address the rows.

The same "col/row" notation is used, except that the column numbers are written with two digits. For example, 04/1 represents the 8-bit byte 01000001, whereas 4/1 represents the 7-bit byte 1000001. Figure 2 shows the 8-bit code table.

2.2 Code Extension Techniques Vocabulary

2.2.1 C0 Sets. C0 sets are repertories of 32 control characters intended to occupy columns 0 and 1 of a 7-bit code chart, or columns 00 and 01 of an 8-bit code chart. The binding of the VDM to character codes uses the same C0 set as is used in ASCII (ANSI X3.4) and the IRV (International Reference Version of ISO 646). Other C0 sets could be used with this VDM binding provided they have the following control characters in the same code chart positions as they occupy in ASCII and the IRV: SO (SHIFT OUT), SI (SHIFT IN), ESC (ESCAPE), RS or IS2 (INFORMATION SEPARATOR TWO), and US or IS1 (INFORMATION SEPARATOR ONE).

2.2.2 C1 Sets. C1 sets are repertories of 32 additional controls intended to occupy columns 08 and 09 of an 8-bit code chart, or to be invoked by special 2-byte escape sequences in a 7-bit environment. This binding of the VDM to character codes does not use any C1 controls. All C1 controls are reserved for future standardization.

2.2.3 G-Sets. G-sets are repertories of 94- or 96-bit combinations. In a 7-bit environment, they occupy columns 2 through 7 of the code chart. In an 8-bit environment, they can occupy either columns 02 through 07 (the "GL" part of the chart) or columns 10 through 15 (the "GR" part). This binding of the VDM to character codes uses a special 96-byte G-set representing VDM functions.

3. Designating and Invoking the C0 Set and VDM G-Set

There are three ways to designate the VDM C0 set and G-set, and to invoke them into the 7-bit or 8-bit code table.

- Implicit designation and invocation.

- Graphics coding system escape sequences.

3.1 Implicit Designation and Invocation. The VDM C0 set and G-set may be invoked implicitly, by agreement between the interchanging parties. Such implicit designation and invocation is suitable only if there is not to be any interchange with services using other code tables.

In a 7-bit environment, the metafile generator and the metafile interpreter would both assume that the C0 set occupies columns 0 and 1 of the code table, that the G-set of VDM functions occupies columns 2 through 7 of the code table. Figure 3 shows such a 7-bit code table.

In an 8-bit environment, the metafile generator and the metafile interpreter would both assume that the C0 set occupies columns 00 and 01 of the 8-bit code table, and that the VDM G-set is a G3 set. They would either both assume that the VDM G-set occupies columns 02 through 07, or else they would both assume that the VDM G-set occupies columns 10 through 15. Figure 4 shows two such 8-bit code tables.

3.2 *Designation and Invocation* as a Graphics Coding System Although the VDM character *coded* graphics binding does not conform to ISO 2022 since sequences of bytes in columns 2 through 7 represent graphic functions rather then graphic characters, the structure of the C and G sets is the same. This permits VDM encoded information to be passed along by the hardware and software that make the usual assumptions about the meaning of certain bit combinations in the C0 set. The VDM coding may be designated and invoked from ISO Character Coding as a Graphics Coding System.

In a 7-bit or 8-bit environment, the VDM coding system is designated and invoked by the escape sequence ESC 2/5 F. Here, F is a final character to be assigned by the International Registration Authority in accordance with procedures specified in ISO International Standard 2375. This escape sequence invokes the ASCII (or IRV) C0 set into columns 0 and 1 of the 7-bit code table. It also designates and invokes the VDM 96-code-position G-set into columns 2 through 7 of the 7-bit code table.

In the 8-bit environment, bit 8 is zero However, an exception exists in the TEXT and APPEND TEXT elements within which ISO 2022 controls may be used. (See Section 8.35.)

The escape sequence, ESC 2/5 4/0, is used by the VDM Graphics Coding system to return to the coding system of ISO 2022. (This escape sequence is specified in DIS2022-1983.) ESC 2/5 4/0 restores the state of the ISO 2022 coding system to that at the time of invocation of the VDM graphics coding system.

4. The VDM C0 Set and G-Set

4.1 The C0 Set. The C0 set occupies columns 0 and 1 (or 00 and 01) of the code chart. Only SO, SI, ESC, IS2, and IS1 are actually used in the metafile. The codes 0/0 (NUL) and 0/8 through 0/13 (HT through CR) may occur within metafile elements, but their effect is not standardized. The meanings of other C0 control characters, should they occur within the metafile, are reserved for future standardization.

Other C0 sets may be used, provided the following conditions are met:

- The bit combinations 0/14, 0/15, 1/11, 1/14, and 1/15 are assigned SO, SI, ESC, IS2, and IS1, respectively.
- Bit combinations 0/0 and 0/8 through 0/13 are allowed in the metafile but have no standardized effect.
- Other bit combinations (0/1 through 0/7, 1/0 through 1/10, 1/12, 1/13) are reserved for future standardization.

Table 1: C0 Control Set

Column_0		Column_1	
0/0	NUL (no standardized effect)	1/0	DLE (reserved)
0/1	SOH (reserved)	1/1	DC1 (reserved)
0/2	STX (reserved)	1/2	DC2 (reserved)
0/3	ETX (reserved)	1/3	DC3 (reserved)
0/4	EOT (reserved)	1/4	DC4 (reserved)
0/5	ENQ (reserved)	1/5	NAK (reserved)
0/6	ACK (reserved)	1/6	SYN (reserved)
0/7	BEL (reserved)	1/7	ETB (reserved)
0/8	BS (no standardized effect)	1/8	CAN (reserved)
0/9	HT (no standardized effect)	1/9	EM (reserved)
0/10	LF (no standardized effect)	1/10	SUB (reserved)
0/11	VT (no standardized effect)	1/11	ESC (ESCAPE)
0/12	FF (no standardized effect)	1/12	IS4 (reserved)
0/13	CR (no standardized effect)	1/13	IS3 (reserved)
0/14	SO (SHIFT OUT)	1/14	IS2 (INFORMATION SEPARATOR TWO)
0/15	SI (SHIFT IN)	1/15	IS1 (INFORMATION SEPARATOR ONE)

4.2 The G-Set of VDM Functions. The G-set of VDM functions occupies 96 contiguous positions in columns 2 through 7 of the 7-bit or 8-bit code chart.

5. Coding the Metafile Elements

Each metafile element is coded as a sequence of bit combinations from the 96-byte G-set of VDM functions.

The first byte is an opcode identifying the particular metafile element.

Subsequent bytes represent parameters and should come from the VDM G-set. If necessary, the IS1 control character (from the C0 set) may be used to separate parameters. (If "character substitution" is in effect, the IS1 may be replaced by a 2-byte sequence of VDM G-set characters.)

The metafile element ends with the IS2 control character from the C0 set. (This control character is sometimes called RS, Record Separator. If "character substitution" is in effect, the IS2 may be replaced by a 2-byte sequence of VDM G-set characters.)

Only characters from the VDM G-set are recognized as opcodes for metafile elements. Also, with one exception, only characters from the VDM G-set and the IS1 information separator control, are recognized as valid parameter characters within a metafile element.

The exception is that G-sets, other than the VDM G-set, are permitted within the string parameters of the TEXT and APPEND TEXT elements. ISO 2022 controls (both control characters and escape sequences) may be used to designate and invoke such G-sets. However, the use of ISO 2022 controls requires prior agreement and care should be taken to return to the VDM G-set for subsequent metafile elements.

Table 2 lists the opcode assignments for the VDM elements. They have been organized as follows: metafile descriptor and control elements are in column 2, primitives are in column 3, attribute elements are in columns 4 and 5, and column 7 includes VDM escape. Column 6 is reserved for future standardization.

Table 2: Opcodes for Metafile Elements

Opcode	7-Bit Coding	8-Bit Coding
METAFILE DESCRIPTOR opcode	2/0	02/0
PICTURE DESCRIPTOR opcode	2/1	02/1
BEGIN-END METAFILE opcode	2/2	02/2
BEGIN PICTURE opcode	2/3	02/3
BEGIN PICTURE BODY opcode	2/4	02/4
END PICTURE opcode	2/5	02/5
INTEGER VDC PRECISION opcode	2/6	02/6
REAL VDC PRECISION opcode	2/7	02/7
VDC EXTENT opcode	2/8	02/8
CLIP RECTANGLE opcode	2/9	02/9
CLIP INDICATOR opcode	2/10	02/10
LOCAL BACKGROUND COLOR opcode	2/11	02/11
MESSAGE opcode	2/12	02/12
APPLICATION DATA opcode	2/13	02/13
(Reserved for standardization)	2/14	02/14
(Reserved for standardization)	2/15	02/15
POLYMARKER opcode	3/0	03/0
POLYLINE opcode	3/1	03/1
POLYGON opcode	3/2	03/2
CELL ARRAY opcode	3/3	03/3
TEXT or APPEND TEXT opcode	3/4	03/4
CIRCLE opcode	3/5	03/5
ARC opcode	3/6	03/6
ARC CLOSE opcode	3/7	03/7
(Reserved for standardization)	3/8	03/8
(Reserved for standardization)	3/9	03/9
(Reserved for standardization)	3/10	03/10
(Reserved for standardization)	3/11	03/11
(Reserved for standardization)	3/12	03/12
(Reserved for standardization)	3/13	03/13
(Reserved for standardization)	3/14	03/14
(Reserved for standardization)	3/15	03/15
POLYMARKER BUNDLE INDEX opcode	4/0	04/0
MARKER TYPE opcode	4/1	04/1
MARKER SIZE opcode	4/2	04/2
MARKER COLOR opcode	4/3	04/3
POLYLINE BUNDLE INDEX opcode	4/4	04/4
LINE TYPE opcode	4/5	04/5
LINE WIDTH	4/6	04/6
LINE COLOR opcode	4/7	04/7
FILL AREA BUNDLE INDEX opcode	4/8	04/8
INTERIOR STYLE opcode	4/9	04/9
FILL COLOR opcode	4/10	04/10
HATCH INDEX opcode	4/11	04/11
PATTERN INDEX opcode	4/12	04/12
PATTERN SIZE opcode	4/13	04/13
PATTERN REFERNECE POINT opcode	4/14	04/14

PATTERN TABLE opcode	4/15	04/15
PERIMETER TYPE opcode	5/0	05/0
PERIMETER WIDTH opcode	5/1	05/1
PERIMETER COLOR opcode	5/2	05/2
TEXT BUNDLE INDEX opcode	5/3	05/3
TEXT FONT INDEX opcode	5/4	05/4
TEXT PRECISION opcode	5/5	05/5
CHARACTER EXPANSION FACTOR opcode	5/6	05/6
CHARACTER SPACING opcode	5/7	05/7
TEXT COLOR opcode	5/8	05/8
CHARACTER HEIGHT opcode	5/9	05/9
CHARACTER ORIENTATION opcode	5/10	05/10
CHARACTER PATH opcode	5/11	05/11
TEXT ALIGNMENT opcode	5/12	05/12
CHARACTER SET INDEX opcode	5/13	05/13
COLOR TABLE opcode	5/14	05/14
SET ASPECT SOURCE FLAGS opcode	5/15	05/15
(Reserved for standardization)	6/0	06/0
(Reserved for standardization)	6/1	06/1
(Reserved for standardization)	6/2	06/2
(Reserved for standardization)	6/3	06/3
(Reserved for standardization)	6/4	06/4
(Reserved for standardization)	6/5	06/5
(Reserved for standardization)	6/6	06/6
(Reserved for standardization)	6/7	06/7
(Reserved for standardization)	6/8	06/8
(Reserved for standardization)	6/9	06/9
(Reserved for standardization)	6/10	06/10
(Reserved for standardization)	6/11	06/11
(Reserved for standardization)	6/12	06/12
(Reserved for standardization)	6/13	06/13
(Reserved for standardization)	6/14	06/14
(Reserved for standardization)	6/15	06/15
(Reserved for standardization)	7/0	07/0
(Reserved for standardization)	7/1	07/1
(Reserved for standardization)	7/2	07/2
(Reserved for standardization)	7/3	07/3
(Reserved for standardization)	7/4	07/4
(Reserved for standardization)	7/5	07/5
(Reserved for standardization)	7/6	07/6
(Reserved for standardization)	7/7	07/7
(Reserved for standardization)	7/8	07/8
(Reserved for standardization)	7/9	07/9
(Reserved for standardization)	7/10	07/10
(Reserved for standardization)	7/11	07/11
(Reserved for 2-char. std. opcodes)	7/12	07/12
VDM ESCAPE opcode	7/13	07/13
(Reserved; not to be used)	7/14	07/14
(Reserved; not to be used)	7/15	07/15

Since the Metafile Descriptor elements appear only once in the metafile, these elements do not have to be as efficiently encoded as, for example, POLYLINE. They are represented by a 2-byte sequence - the primary opcode 2/0 and a secondary opcode which indicates the specific element. The values of the secondary opcodes are described in Table 3.

Table 3: Secondary Opcode Values for Metafile Descriptor Elements

VDM VERSION opcode	3/0	03/0
VDM DESCRIPTION opcode	3/1	03/1
VDC TYPE opcode	3/2	03/2
MAXIMUM COLOR INDEX opcode	3/3	03/3
NON-VDC INTEGER PRECISION opcode	3/4	03/4
NON-VDC REAL PRECISION opcode	3/5	03/5
INDEX PRECISION opcode	3/6	03/6
COLOR PRECISION opcode	3/7	03/7
COLOR INDEX PRECISION opcode	3/8	03/8
VDM DEFAULTS REPLACEMENT opcode	3/9	03/9
VDM ELEMENT LIST opcode	3/10	03/10
CHARACTER SET LIST opcode	3/11	03/11
FONT LIST opcode	3/12	03/12

The similar technique has been used for the Picture Descriptor (PD) elements with the primary opcode 2/1 and the secondary opcodes as described in Table 4.

Table 4: Secondary Opcode Values for Picture Descriptor Elements

SCALING MODE opcode	3/0	03/0
COLOR SPECIFICATION MODE opcode	3/1	03/1
LINEWIDTH SPECIFICATION MODE opcode	3/2	03/2
MARKER SIZE SPECIFICATION MODE opcode	3/3	03/3
PERIMETER WIDTH SPECIFICATION MODE opcode	3/4	03/4

The elements BEGIN METAFILE and END METAFILE are represented by the primary opcode 2/2. The secondary opcode for BEGIN METAFILE is 3/0 and END METAFILE is 3/1. The TEXT and APPEND TEXT elements are represented by the primary opcode 3/4.

6. Character Substitution

To accommodate systems in which it is convenient to include C0 control characters, the SPACE character (2/0), or the DELETE character (7/15) in the metafile, this VDM binding includes a "character substitution" option. Characters in the range of 0/0 to 1/15, the characters 2/0, 7/14, and 7/15 may be replaced by 2-byte sequences from the VDM G-set provided each such 2-byte sequence is declared in the first parameter of the BEGIN METAFILE element.

Table 5 shows the characters that may be replaced by such 2-byte sequences, and the 2-byte sequences with which they are replaced. Note that all the 2-byte sequences begin with 7/14. Therefore, if the character substitution option is used at all, that character (7/14) must be declared as one of the characters that is being replaced with a 2-byte sequence. This is done by including its replacement sequence (7/14 3/14) in the first parameter of BEGIN METAFILE.

Once a character (or rather, its 2-byte replacement) has been declared in the BEGIN METAFILE element, "character substitution" is in effect for that character and will remain in effect until the end of the metafile (that is, until the END METAFILE element). The metafile interpreter ignores any characters for which character substitution is in effect.

Example:

It is desirable to avoid using the characters IS2, IS1, SPACE, TILDE, and DELETE in the metafile, and for the metafile interpreter to ignore those characters if they are inadvertently inserted (for example, by a host operating system or some process other than the metafile generator).

In that case, the metafile generator declares "character substitution" for the above four characters and for the TILDE character, 7/14. It does this in the first parameter of BEGIN METAFILE as follows:

```
BEGIN METAFILE
  * 2/2 3/0                          (BEGIN METAFILE opcode)
    7/14 5/14 7/14 5/15              ( string: ~^ ~_~`~>~? )
    7/14 6/0 7/14 3/14 7/14 3/15
    7/14 5/15                        { substitute for IS1 to
                                       terminate string }

     string: metafile name
    7/14 5/14                        { substitute for IS2 to
                                       terminate command }
```

Throughout the metafile, wherever the metafile generator would otherwise put an IS1, IS2, SPACE, DELETE, or TILDE (7/14) character, it substitutes the 2-byte sequences 7/14 5/14, 7/14 5/15, 7/14 6/0, 7/14 3/14, or 7/14 3/15, respectively. (This character substitution occurs even for the IS1 and IS2 within the BEGIN METAFILE element itself.)

Upon interpretation, wherever the metafile interpreter encounters the character 7/14, it interprets it as the first character of a 2-byte sequence representing one of these characters. The metafile interpreter ignores the characters 1/14 (IS2), 1/15 (IS1), 2/0 (SPACE), and 7/15 (DELETE).

Table 5: Character Substitution

The character:	May be replaced with: 7-bit	8-bit	ASCII char.
0/0 (NUL)	7/14 4/0	07/14 04/0	(~@)
0/1 (SOH)	7/14 4/1	07/14 04/1	(~A)
0/2 (STX)	7/14 4/2	07/14 04/2	(~B)
0/3 (ETX)	7/14 4/3	07/14 04/3	(~C)
0/4 (EOT)	7/14 4/4	07/14 04/4	(~D)
0/5 (ENQ)	7/14 4/5	07/14 04/5	(~E)
0/6 (ACK)	7/14 4/6	07/14 04/6	(~F)
0/7 (BEL)	7/14 4/7	07/14 04/7	(~G)
0/8 (BS)	7/14 4/8	07/14 04/8	(~H)
0/9 (HT)	7/14 4/9	07/14 04/9	(~I)
0/10 (LF)	7/14 4/10	07/14 04/10	(~J)
0/11 (VT)	7/14 4/11	07/14 04/11	(~K)
0/12 (FF)	7/14 4/12	07/14 04/12	(~L)
0/13 (CR)	7/14 4/13	07/14 04/13	(~M)
0/14 (SO)	7/14 4/14	07/14 04/14	(~N)
0/15 (SI)	7/14 4/15	07/14 04/15	(~O)
1/0 (DLE)	7/14 5/0	07/14 05/0	(~P)
1/1 (DC1)	7/14 5/1	07/14 05/1	(~Q)
1/2 (DC2)	7/14 5/2	07/14 05/2	(~R)
1/3 (DC3)	7/14 5/3	07/14 05/3	(~S)
1/4 (DC4)	7/14 5/4	07/14 05/4	(~T)
1/5 (NAK)	7/14 5/5	07/14 05/5	(~U)
1/6 (SYN)	7/14 5/6	07/14 05/6	(~V)
1/7 (ETB)	7/14 5/7	07/14 05/7	(~W)
1/8 (CAN)	7/14 5/8	07/14 05/8	(~X)
1/9 (EM)	7/14 5/9	07/14 05/9	(~Y)
1/10 (SUB)	7/14 5/10	07/14 05/10	(~Z)
1/11 (ESC)	7/14 5/11	07/14 05/11	(~[)
1/12 (FS or IS4)	7/14 5/12	07/14 05/12	(~\)
1/13 (GS or IS3)	7/14 5/13	07/14 05/13	(~])
1/14 (RS or IS2)	7/14 5/14	07/14 05/14	(~^)
1/15 (US or IS1)	7/14 5/15	07/14 05/15	(~_)
2/0 (SPACE)	7/14 6/0	07/14 06/0	(~`)
7/14 (TILDE,~)	7/14 3/14	07/14 03/14	(~>)
7/15 (DELETE)	7/14 3/15	07/14 03/15	(~?)

In a 7-bit environment or in an 8-bit environment in which the VDM G-set is invoked into the "GL" part of the code table (columns 02 through 07), Table 5 may be summarized as follows: "Each character from decimal 0 (0/0) to decimal 32 (2/0) may be replaced by a 2-byte sequence in which the first byte is decimal 126 (7/14) and the decimal equivalent of the second byte is obtained by adding 64, modulo 128, to the decimal equivalent of the original character."

7. Coding the Parameters of Metafile Elements

7.1 Coding Integers. Integers are coded as sequences of bytes in the range from 2/0 to 7/15.

If a character is from column 2 or 3 (10 or 11) of the chart, it is the last character in the integer's coded representation. In that case, bits b1 to b4 are the least-significant four bits in the integer's sign-magnitude binary representation. Bit b5 is one sign bit: 1 if positive, 0 if negative. The format is as follows:

If a character is from columns 4 through 7 (12 through 15) of the code chart, it contains additional, more significant bits of the binary numeral. The most significant bits are sent in the first such character, less significant bits in subsequent characters. The format is as follows:

The "X" is the parity bit (or omitted bit) in a 7-bit environment. In an 8-bit environment, it is either 0 or 1, according to whether the G-set is invoked into the left or right half of the 8-bit code table. "s" is the sign bit, and "bbb..." are bits representing the magnitude of the integer.

Integers in the range of -15 to +15 can be coded as single bytes.

(integer: +1) = 3/1 (integer: -1) = 2/1
(integer: +15) = 3/15 (integer: -15) = 2/15

Larger integers require more bytes.

(integer: +16) = 4/1 3/0 (integer: -16) = 4/1 2/0
(integer: +1034) = 4/1 4/0 3/10 (integer: -1034) = 4/1 4/0 2/10

The number zero must always be coded as "plus zero" 3/0. The "minus zero" coding is reserved for future standardization.

Any integer can be coded with leading most-significant-bits of all zero. For example, 3/3 and 4/0 4/0 3/3 are both valid codings for integer: +3; however, efficient metafile generators

should avoid such redundant codings.

The size of integer parameters is limited by the current NON-VDC INTEGER PRECISION value.

7.2 Coding Real Numbers. Each real number is coded as an integral mantissa followed by an optional exponent. One of the bits in the last byte of the mantissa tells whether the exponent follows. The exponent is the power of two by which the integral mantissa is to be multiplied. If the exponent is omitted, it assumes a default value of zero. The exponent is coded as an integer value.

The mantissa begins with zero or more characters from columns 4 through 7 of the code chart. Each such character holds six of the mantissa's more significant bits. The format is as follows:

```
b8              b1
+-+-+------------+
|X|1|b b b b b b|
+-+-+------------+
```

The mantissa ends with a terminator character from column 2 or 3 (10 or 11) of the code chart. In this terminator character, bits b7 and b6 are 0 and 1; bit b5 is 1 if an exponent follows the mantissa. 0 if there is no exponent following the mantissa; bit b4 is 0 for negative mantissas, 1 for non-negative mantissas; and bits b3 to b1 are the three least-significant bits in the magnitude of the mantissa. The format is as follows:

```
b8              b1
+-+---+-+-------+
|X|0 1|s|s|b b b|
+-+---+-+-------+
```

Mantissas or exponents of "minus zero" are not allowed. Those codings are reserved for future standardization.

For example, in a 7-bit environment the binary numeral +1.110010110011 is coded as follows:

(real: binary +001.110 010110 011)

- mantissa: +001110 010110 011, "exponent-follows"
  exponent: -12

- more-significant-bits: "001110"
  more-significant-bits: "010110"
  "1", positive sign, "011"
  integer: -12


```
|    byte 1    |    byte 2    |    byte 3    |    byte 4    |
+-+------------+-+------------+---+-+-+-----+---+-+--------+
- |1|0 0 1 1 0|1|0 1 0 1 1 0|0|0 1|1|1|0 1 1|0 1|1|0 1|1|0|1 1 0 0|
+-+------------+-+------------+---+-+-+-----+---+-+--------+
|            |              |              |              |
-      4/14          5/6           3/11          2/12
```

If the exponent is omitted from a real data type, a default exponent of zero is assumed. The size of real parameters is limited by the current NON-VDC REAL PRECISION value.

7.3 Coding VDCs and Points. A point is a pair of VDC scalars. A VDC scalar is either an integer or real number according to whether VDC TYPE is integer or real.

When VDC TYPE is integer, the encodings of the VDC and point data types are as described in Section 7.1, Coding Integers. The size of the VDC and point parameters is limited by the current INTEGER VDC PRECISION value. When VDC TYPE is real, the encodings of the VDC and point data types are as described in Section 7.2, Coding Real Numbers. The size of the VDC and point parameters is limited by the current REAL VDC PRECISION value. If the exponent is omitted from a VDC value, a default exponent of zero is assumed. If the exponent is omitted from a point data type, a default exponent is assumed as follows:

- If the point is the first point of a point list, the omitted exponent is deemed to be zero.

- If the point is not the first point in the point list, an omitted exponent may assume a value other than zero. If the exponent is omitted from the x-component of a real point, it defaults to the value of the exponent in the preceding *x-component*. Similarly, an *exponent* omitted from a *y-component* assumes the value of the preceding point's y-component.

7.4 Coding Point List Parameters. The POLYLINE, POLYMARKER, and POLYGON elements have point list parameters. In these point list parameters, the first point in the list is an absolute position in VDC space. Subsequent points in the list are each specified as a displacement from the preceding point.

If the IS1 control character occurs within a POLYLINE element, it means "end this polyline and start another." Similarly, when IS1 occurs within a POLYGON element, it means "end this polygon and start another." Again, should IS1 ever occur within a POLYMARKER element, it means "end this polymarker and start another." Point list parameters following IS1 follow the rules of point lists, that is, the first point in the list is an absolute position and subsequent points are specified as displacements.

7.5 Color Specifiers. Color specifiers are coded either as color indexes (if COLOR SELECTION MODE is 'indexed') or as RGB parameters (if COLOR SELECTION MODE is 'direct').

Color indexes are coded as integers.

The RGB parameters used if COLOR SELECTION MODE is 'direct' are coded as a series of bytes in the range of 4/0 to 7/15. The six least significant bits in each byte hold binary bits representing the red, green, and blue color values. If COLOR PRECISION is set to N bits, there are 3N color value bits; there are only as many bytes in an RGB parameter as are necessary to hold those 3N bits.

For example, if COLOR PRECISION is set to 5 bits, RGB parameters have the following form:

The first five color precision bits (bits b6 to b2 of the first byte) hold the red color value, expressed as a 5-bit binary numeral. The next five color precision bits (bit b1 of the first byte and bits b6 to b3 of the second byte) hold the green color value, again as a 5-bit binary numeral. The third group of five bits (bits b2 and b1 of the second byte, and bits b6 to b4 of the third byte) hold the blue color value. The leftover bits (bits b3 to b1 of the third byte) should be zero and are ignored by the metafile interpreter.

7.6 Color Index Lists. The color index list of CELL ARRAY needs to be coded compactly. It contains color indexes for dx*dy cells of the cell array. If there are any excess color indexes, they are ignored (that is, once the metafile interpreter encounters the dxdy'th color index, it ignores the remainder of the color index list). The metafile interpreter resumes processing after encountering I32, which terminates the picture element.

There are two ways to code the color index list. If many adjacent cells have the same color index, runlength coding is efficient. However, for short runs of only one or two cells, it is better to send the color indexes as a stream of bits.

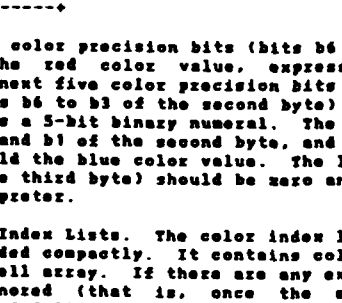
7.6.1 Runlength-Coded Sublists. If many adjacent color cells have the same color index, an efficient metafile generator would code them using runlength codes. For each run, the color of the cells is specified, followed by a count telling how many cells are in the run.

The runlength-coded sublist is introduced by the 2/1 character for a 7-bit code or the 02/1 character for an 8-bit code.

7.6.2 Bit-Stream-Coded Sublists. If adjacent color cells are not the same color, runlength coding is inappropriate. In that case, the metafile generator can use bit-stream-coded sublists, in which the color indexes are packed as tightly as possible, six bits at a time, into characters from columns 4 through 7 of the code chart (possibly columns 12 through 15 in an 8-bit environment). The bit-stream-coded sublist is introduced by 2/2 in a 7-bit code, 02/2 in an 8-bit code.

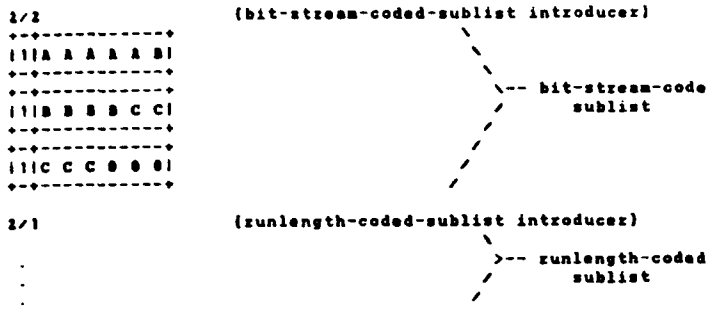
The bits in the bit stream represent color indexes. For example, suppose that COLOR INDEX PRECISION is 5 so that each color index has 5 bits. Suppose further that we are using a 7-bit code. Then, the color indexes are packed into the bit stream as follows:

bit-stream-coded-sublist = 2/2 (sublist introducer)



Note that AAAAA is the binary numeral for the first color-indexed, BBBBB is the binary numeral for the second color-index, and so on.

In the last byte of a bit-stream-coded sublist, any unused bits are ignored. For example:



The first four bytes comprise a bit-stream-coded sublist. The first byte, 2/2, introduces the sublist. The following three bytes hold color indexes AAAAA, BBBBB, and CCCCC for three pixels.

The next byte is from column 2 of the chart, so it terminates the bit-stream-coded sublist. Since it is 2/1 (the code that introduces a runlength-coded sublist), subsequent bytes will be interpreted as color indexes and runlength codes.

The leftover bits, in the last byte of the bit-stream-coded sublist, should be zero and are ignored.

If the leftover bits are the right number of bits to comprise a valid color index, they are treated as a valid color index. That is, they are not ignored (unless the count of duªdy color indexes has been exhausted).

7.7 String Parameters. Strings are coded as sequences of VDM G-set characters in the range of 2/0 to 7/14 (in an 8-bit environment, 02/0 to 07/14). The bit combinations 0/0, 0/8 to 0/13, and 7/15 (or 07/15) may also occur within string parameters, but they have no standardized effect.

If a string parameter is not the last parameter in a metafile element, it is terminated with an IS1 control character. If the string parameter is the last parameter of the metafile element, the IS1 control is not necessary; the IS2 that terminates the metafile element also terminates the string parameter.

The string parameters that occur within the TEXT and APPEND TEXT elements may include characters from G-sets other than the VDM G-set, together with ISO 2022 controls to designate and/or invoke other such G-sets and similar ISO 2022 controls to return to the VDM G-set. The effect of using these ISO 2022 controls within the text string is implementation dependent. The ISO 2022 controls to invoke G-sets into the code table are as follows:

(a) The C0 controls, SHIFT OUT and SHIFT IN. SO = 0/14, SI = 0/15

(b) The "locking shift" escape sequences. LS2 = ESC 6/14 = 1/11 6/14; LS3 = ESC 6/15 = 1/11 6/15; LS1R = ESC 7/14 = 1/11 7/14; LS2R = ESC 7/13 = 1/11 7/13; LS3R = ESC 7/12 = 1/11 7/12

(c) The "single shift" controls SINGLE SHIFT TWO and SINGLE SHIFT THREE provided a C0 or C1 set that has such controls as have been designated and invoked.

(d) The ISO 2022 escape sequences to designate G-sets as G0, G1, G2, or G3.

7.8 Enumerated Parameters. Enumerated parameters represent choices within a fixed set of options. They are coded as integers.

7.9 Index and Color Index Parameters. Both indexes and color indexes are coded as integers. Private (nonstandard) values of index parameters are all coded using negative integers.

8. Coding Each Metafile Element

For convenience, the 7-bit opcode is given for each element described in the following sections. To determine the 8-bit opcode, a zero is added in front of the column specification (for example, 06/1 instead of 6/1). A list of opcodes is given in Table 2.

Notation Used:

<symbol>* = 0 or more occurrences

<symbol>+ = 1 or more occurrenc

<symbol>o = optional, 0 or more occurrences

{comment} = explanation of a production

<x:y> = construct x with meaning y

8.1 VDM VERSION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<VDM-VERSION-opcode: 3/0>
<integer>
<IS2>

8.2 VDM DESCRIPTION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<VDM-DESCRIPTION-opcode: 3/1>
<string: description-of-the-metafile>
<IS2>

8.3 VDC TYPE

<METAFILE-DESCRIPTOR-opcode: 2/0>
<VDC-TYPE-opcode: 3/2>
<enumerated: VDC-type>
<IS2>

<enumerated:VDC-type> = <integer: 0> {integer points}
| <integer: 1> {real points}

8.4 NON-VDC INTEGER PRECISION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<NON-VDC-INTEGER-PRECISION-opcode: 3/4>
<integer: largest-integer-code + 1 >
<IS2>

The largest-integer-code tells how many bits occur in the largest possible magnitude for an integer. For example, if integers in the metafile can range from -32767 to +32767, the largest-integer-code is 15. One additional bit is required for the sign, and so is added to obtain the proper precision.

8.5 NON-VDC REAL PRECISION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<NON-VDC-REAL-PRECISION-opcode: 3/5>
<integer: largest-real-code+1>
<integer: smallest-real-code>
<IS2>

The largest-real-code tells how many bits can occur to the left of the (binary) radix point in the largest possible real number. For example, if the largest possible real number is binary +111111111111111111111.110, the largest-real-code would be 20. One additional bit is required for the sign, and so is added to obtain the appropiate field width.

The smallest-real-code shows how small a nonzero real number can be. For example, if the smallest magnitude a nonzero real number can have is 1/64 (binary 0.000001), the smallest-real-code is -6, indicating 6 bits of precision to the right of the binary radix point.

It is possible for smallest-integer-code to be positive. For example, suppose the largest possible positive number is binary +1111111111111111000.0 and the smallest possible positive number is binary +1000.0 (decimal 8). In that case, the largest-integer-code is +20, and the smallest-integer-code is +3.

The largest-real-code must be greater than the smallest-real-code. The difference (largest-real-code plus one(for sign) minus smallest-real-code) tells how many bits of precision must be carried when performing arithmetic with real numbers.

8.6 INDEX PRECISION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<INDEX-PRECISION-opcode: 3/6>
<integer: maximum-number-of-bits>
<IS2>

8.7 COLOR PRECISION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<COLOR-PRECISION-opcode: 3/7>
<integer: number-of-bits>
<IS2>

The parameter number-of-bits tells how many binary bits are used to specify the red, green, and blue components of color mixtures.

8.8 COLOR INDEX PRECISION

<METAFILE-DESCRIPTOR-opcode: 2/0>
<COLOR-INDEX-PRECISION-opcode: 3/8>
<integer: number-of-bits>
<IS2>

The parameter number-of-bits specifies the number of bits used in the binary numerals that represent color indexes. For example, if number-of-bits is 3, there are 8 possible color indexes: binary 000 (decimal 0) to binary 111 (decimal 7).

8.9 MAXIMUM COLOR INDEX

<METAFILE-DESCRIPTOR-opcode: 2/0>
<MAXIMUM-COLOR-INDEX-opcode: 3/3>
<integer: maximum-color-index>
<IS2>

8.10 VDM ELEMENT LIST

<METAFILE-DESCRIPTOR-opcode: 2/0>
<VDM-ELEMENT-LIST-opcode: 3/10>
<opcode-character>+
<IS2>

The <opcode-character>s are the opcodes for any VDM element to be included in the metafile.

8.11 VDM DEFAULT REPLACEMENT

```
<METAFILE-DESCRIPTOR-opcode: 2/0>
<VDM-DEFAULTS-opcode: 3/9> <element-that-sets-a-default>
(<IS1> <element-that-sets-a-default>)*
<IS2>
```

The <element-that-sets-a-default> list is given in Part 1, Section 6. The coding format for each of the elements that appear in VDM DEFAULTS REPLACEMENT is exactly as described in this section. Coding Each Metafile Element. The parameter values express the default values to be used. For example, in a 7-bit encoding, the default values for CLIP INDICATOR and MATCH INDEX would be as follows:

```
2/0           (METAFILE-DESCRIPTOR-opcode)
3/9           (VDM-DEFAULTS-opcode)
2/10          (CLIP-INDICATOR-opcode)
integer: 0    (clip-indicator-on)
IS1           (IS1-terminates first parameter)
4/11          (MATCH-INDEX-opcode)
integer: 2    (default index equal to 2)
IS2           (IS2-terminates VDM-DEFAULTS)
```

8.12 FONT LIST

```
<METAFILE-DESCRIPTOR-opcode: 2/0>
<FONT-LIST-opcode: 3/12>
<font-declaration>
(<IS1> <font-declaration>)*
<IS2>

<font-declaration> = <string: name-of-font>
```

The FONT LIST element declares the character fonts that may be named in subsequent TEXT FONT INDEX elements and establishes the font index value that is associated with each such character font.

The first <font-declaration> in the list names the character font whose font index value is to be 1. Likewise, the second, third, fourth, etc., <character-font-declaration>s name the fonts whose index values are to be 2, 3, 4, etc. The <font-declaration>s are separated from each other by <IS1> control characters.

(An index value of 1 always refers to the default font -- the font that would be used if there were no TEXT FONT INDEX element in the metafile.)

8.13 CHARACTER SET LIST

```
<METAFILE-DESCRIPTOR-opcode: 2/0>
<CHARACTER-SET-LIST-opcode: 3/11>
<character-set-declaration>
(<IS1> <character-set-declaration>)*
<IS2>

<character-set-declaration>
   = <enumerated: character-set-type>
     <string: end-of-escape-sequence-to-designate-character
              -set>

<enumerated: character-set-type>
   = <integer: 0>  (to declare a 94-character G-set)
   | <integer: 1>  (to declare a 96-character G-set)
   | <integer: 2>  (to declare a multibyte 94-character G-set)
   | <integer: 3>  (to declare a multibyte 96-character G-set)
   | <integer: 4>  (to declare a "complete code" char. set)
```

The CHARACTER SET LIST element declares the character sets that can be named in subsequent CHARACTER SET INDEX elements and establishes the character set index value that is associated with each of these character sets.

The first <character-set-declaration> in the list names the character set whose character set index value is to be 1. Likewise, the second, third, fourth, etc., <character-set-declaration>s name the character sets whose index values are to be 2, 3, 4, etc. The <character-set-declaration>s are separated from each other by <IS1> control characters.

(An index value of 1 always refers to the default character set--the character set that would be used if there were no previous CHARACTER SET INDEX elements in the metafile.)

Each <character-set-declaration> has two parts an <integer> and a short <string>. The <integer> specifies which type of character set is being declared (that is, which type of ISO 2022 designating escape sequence is associated with that character set). The <string> consists of the character that forms the "tail end" of such designating escape sequences for that character set.

There are five types of character sets: 94-character G-sets, 96-character G-sets, 94-character multibyte G-sets 96-character multibyte G-sets, and character sets intended to be designated as "complete codes".

8.13.1 94-CHARACTER G-SETS. These character sets are designated by ISO 2022 escape sequences of the form <ESC> <I> <I>(0) <F> Here, <I> is either 2/8, 2/9, 2/10, or 2/11, <I>(0) represents zero or more intermediate characters from

column 2 of the code chart; and <F> is a final character from columns 3 through 7 of the code chart. If <F> is from column 3 of the code chart, the character set is a "private" character set. If <F> is from columns 4 through 7 of the code chart, the character set is a "standard" character set in the sense that it and its designating escape sequences are registered in the International Register Of Character Sets To Be Used With Escape Sequences.

For 94-character G-sets, the <character-set-declaration> consists of <integer: 0>, followed by a string consisting of all characters in the ISO 2022 designating escape sequence except the first two characters, <ESC> <I1>.

For example, the G-set from the U.K.'s national 7-bit character is registered in the International Register Of Character Sets To Be Used With Escape Sequences. Its designating escape sequences are as follows:

<ESC> 2/8 4/1 (to designate it as G0)
<ESC> 2/9 4/1 (to designate it as G1)
<ESC> 2/10 4/1 (to designate it as G2)
<ESC> 2/11 4/1 (to designate it as G3)

Again, the French character set (1982 version, from the 1982 version of AFNOR NF Z 62-010) is registered in the International Register Of Character Sets To Be Used With Escape Sequences. Its designating escape sequences are as follows:

<ESC> 2/8 6/6 (to designate it as G0)
<ESC> 2/9 6/6 (to designate it as G1)
<ESC> 2/10 6/6 (to designate it as G2)
<ESC> 2/11 6/6 (to designate it as G3)

Therefore, a CHARACTER SET LIST element could specify that the U.K. character set is to be referred to by character set index 1, and the French character set by character set index 2, as follows:

<CHARACTER-SET-LIST: U.K., French>

- <CHARACTER-SET-LIST-opcode>
  <character-set-declaration: U.K. character set> <IS1>
  <character-set-declaration: French character set> IS2>

- <CHARACTER-SET-LIST-opcode>
  <integer: 0> 4/1 <IS1>
  <integer: 0> 6/6 <IS2>

- 6/11 3/0 4/1 1/14 3/0 6/6 1/15

8.13.2 96-CHARACTER G-SETS. These character sets are similar to 94-character G-sets, but include the code positions 2/0 and 7/15, which are excluded from 94-character G-sets. Their ISO 2022 designating escape sequences take the form <ESC> <I1> <I>(0) <F>, where the first intermediate character <I1> is either 2/13, 2/14, or 2/15. The remainder of the escape sequence is similar to the escape sequences for 94-character G-sets: zero or more intermediate characters from column 2 of the code chart and a final character from columns 3 through 7 of the code chart.

For 96-character G-sets, the <character-set-declaration> consists of <integer: 1>, followed by a string consisting of all characters in the ISO 2022 designating escape sequence except the first two characters, <ESC> <I1>.

So far, no 96-character G-sets of graphic characters have been registered in the International Register Of Character Sets To Be Used With Escape Sequences. However, it is possible for interchanging parties to agree on a private 96-character G-set whose designating escape sequences would end with a character from column 3 of the code chart. For example, the following might be private escape sequences to designate such a G-set:

<ESC> 2/13 3/0 (to designate it as G1)
<ESC> 2/14 3/0 (to designate it as G2)
<ESC> 2/15 3/0 (to designate it as G3)

(96-character G-sets may not be designated as G0 sets.)

For example, the following CHARACTER SET LIST element establishes the U.K. 94-character G-set, the French 94-character G-set, and a private 96-character G-set as the character sets named by character set indexes 1, 2, and 3, respectively:

<CHARACTER-SET-LIST: U.K., French, private-96-char-set>

- <CHARACTER-SET-LIST-opcode>
  <character-set-declaration: U.K. character set> <IS1>
  <character-set-declaration: French character set> <IS1>
  <character-set-declaration: private-96-char-set> <IS2>

- <CHARACTER-SET-LIST-opcode>
  <integer: 0> 4/1 <IS1>
  <integer: 0> 6/6 <IS1>
  <integer: 1> 3/0 <IS2>

- 6/11 3/0 4/1 1/14 3/0 6/6 1/14 3/1 3/0 1/15

8.13.3 94-CHARACTER MULTIBYTE G-SETS. A 94-character multibyte G-set can contain 94 to the Nth power characters, each coded as a sequence of N bytes from columns 2 through 8 of the code chart--not including the bytes 2/0 and 7/15, which are excluded from 94-character G-sets. For example, a 94-character 2-byte G-set can contain 8,836 characters.

The ISO 2022 designating escape sequences for 94-character multibyte G-sets have the following forms:

```
<ESC> 2/4 <F>        (to designate it as G0)
<ESC> 2/4 2/9 <F>    (to designate it as G1)
<ESC> 2/4 2/10 <F>   (to designate it as G2)
<ESC> 2/4 2/11 <F>   (to designate it as G3)
```

For 94-character multibyte G-sets, the <character-set-declaration> consists of <integer: 2>, followed by a string consisting only of the final character in the ISO 2022 designating escape sequence.

For example, a Japanese 2-byte character set of 6802 graphic characters has been registered in the International Register Of Character Sets To Be Used With Escape Sequences, and its designating escape sequences have the form shown above, with the final character <F> being 4/0. Thus, the following CHARACTER SET LIST element could be used to specify that this 2-byte Japanese character set is to be referred to by character set index 2:

```
<CHARACTER-SET-LIST: Japanese-2-byte-char-set>

  * <CHARACTER-SET-LIST-opcode>
    <character-set-declaration:
     Japanese-2-byte-char-set>  <IS2>

  * <CHARACTER-SET-LIST-opcode>
    <integer: 2> 4/0 <IS2>

  * 6/11 3/2 4/0 1/15
```

8.13.4 96-CHARACTER MULTIBYTE G-SETS OF GRAPHIC CHARACTERS. A 96-character multibyte G-set is similar to a 94-character multibyte G-set except that it can include the bytes 2/0 and 7/15. Thus, a 96-character 2-byte G-set could have 96 times 96 (or 9216) 2-byte character codes.

The ISO 2022 designating escape sequences for 96-character multibyte G-sets have the following forms:

```
<ESC> 2/4 2/13 <F>   (to designate it as G1)
<ESC> 2/4 2/14 <F>   (to designate it as G2)
<ESC> 2/4 2/15 <F>   (to designate it as G3)
```


It is not possible to designate a 96-character multibyte G-set as a G0 set.

The <character-set-declaration> for a 96-character multibyte G-set consists of <integer: 3> followed by a <string> consisting only of the final character <F> in the character set's ISO 2022 designating escape sequence.

So far, no 96-character multibyte G-sets have been registered in the International Register of Character Sets To Be Used with Escape Sequences.

8.13.5 CHARACTER SETS INTENDED TO BE DESIGNATED AS COMPLETE CODES. Other character sets may not fit the ISO 2022 "G-set" structure. ISO 2022 provides an escape sequence format for invoking coding systems different from ISO 2022. The compete code escape sequences have the following form:

```
<ESC> 2/5 <I>o <F>
```

where <I>o means "zero or more characters from column 2 of the code chart", and <F> is a final character from columns 3 through 7 of the code chart. If <F> is from column 3 the coding system is a private code. If <F> is from columns 4 through 7, it is a code for which a designating and invoking escape sequence has been registered in the International Register Of Character Sets To Be Used With Escape Sequences. If <F> is from column 4, it is a code of fewer than 7 bits; if <F> is from column 5, it is a 7-bit code; if <F> is from column 6, it is an 8-bit code; and if <F> is from column 7, it is a code of more than 8 bits.

The <character-set-declaration> for a character set that would be invoked as a coding system different from ISO 2022 consists of <integer: 4> followed by a <string> consisting only of those characters in the code's ISO 2022 escape sequence which come after the first two characters. <ESC> 2/5

No such coding system has so far been registered. However, private code could be used. For example, suppose the interchanging parties have agreed on a private 8-bit code to be invoked by the following escape sequence:

```
<ESC> 2/5 2/0 3/0
```

The following CHARACTER SET LIST element would declare the French character set to have character set index 1 and that 8-bit private code to have character set index 2:

<CHARACTER-SET-LIST: French, private-coding-system>

- <CHARACTER-SET-LIST-opcode>
  <character-set-declaration: French> <IS1>
  <character-set-declaration:
   private-coding-system> <IS2>

- <CHARACTER-SET-LIST-opcode>
  <integer: 0> 6/6 <IS1>
  <integer: 4> 2/0 3/0 <IS2>

- 6/11 3/0 6/6 1/14 3/4 2/0 3/0 1/15

8.14 BEGIN METAFILE

<BEGIN-or-END-METAFILE-opcode: 2/2>
<BEGIN-METAFILE-opcode: 3/0>
<string: substitution-codes> <IS1>
<string: metafile-name>
<IS2>

The BEGIN-or-END-METAFILE element marks the beginning or the end of a metafile. (More than one metafile can be recorded in a single computer file provided each metafile is delimited with the secondary opcodes BEGIN-METAFILE and END METAFILE.)

The first parameter, <string: substitution-codes>, lists those 2-character substitution codes that will be used in the metafile to represent C0 control characters (bit combinations in the range of 0/0 to 1/15) or the SPACE, TILDE, and DELETE characters (bit combinations 2/0, 7/14, and 7/15, respectively. Character substitution will be in effect only for those bit combinations whose substitution codes are listed in this <string> parameter. If the <string> parameter is empty, character substitution is not to be used in this metafile.

The end of the first parameter is marked by an <IS1> control character (or, if character substitution is in effect for that character, by its 2-character substitution code "-_" or 7/14 6/15).

8.15 END METAFILE

<BEGIN/END-METAFILE-opcode: 2/2>
<END-METAFILE-opcode: 3/1> <IS2>

8.16 BEGIN PICTURE

<BEGIN-PICTURE-opcode: 2/3>
<string: picture-name>
<color-specifier: background-color>
<IS2>

<color-specifier>
  = <integer: color-index>{if COLOR SELECTION MODE is indexed}
  | <RGB> {if COLOR SELECTION MODE is direct}

8.17 BEGIN PICTURE BODY

<BEGIN-PICTURE-BODY-opcode: 2/5>
<IS2>

8.18 END PICTURE

<END-PICTURE-opcode: 2/4> <IS2>

8.19 LOCAL BACKGROUND COLOR

<LOCAL-BACKGROUND-COLOR-opcode: 2/11>
<enumerated: on-off flag>
<color-specifier: local-background-color>
<IS2>

<enumerated: on-off flag> = <integer: 0> {on}
  | <integer: 1> {off}

<color-specifier>
  = <integer: color-index>{if COLOR SELECTION MODE is indexed}
  | <RGB> {if COLOR SELECTION MODE is direct}

8.20 INTEGER VDC PRECISION

<INTEGER-VDC-PRECISION-opcode: 2/6>
<integer: largest-integer-code+1>
<IS2>

The largest-integer-code tells how many bits occur in the largest possible magnitude for an integer point. For example, if integer points in the metafile can range from -32767 to +32767, the largest-integer-code is 15. One additional bit is required for the sign, thus an additional bit is added to obtain the appropiate field width.

8.21 REAL VDC PRECISION

<REAL-VDC-PRECISION-opcode: 2/7>
<integer: largest-real-code+1>
<integer: smallest-real-code>
<IS2>

The largest-real-code tells how many bits can occur to the left of the (binary) radix point in the largest possible real number. For example, if the largest possible real number is binary +1111111111111111111.111, the largest-real-code would be 20. One additional bit is required for the sign, thus an additional bit is added to obtain the appropiate field width.

The smallest-real-code shows how small a nonzero real number can be. For example, if the smallest magnitude a nonzero real number can have is 1/64 (binary 0.000001), the smallest-real-code is -6, which indicates 6-bit precision to the right of the binary radix point.

It is possible for the smallest-integer-code to be positive. For example, suppose the largest possible positive number is binary +1111111111111111000.0 and the smallest possible positive number is binary +1000.0 (decimal 8). In that case, the largest-integer-code is +20, and the smallest-integer-code is +3.

The largest-real-code must be greater than the smallest-real-code. The difference (largest-real-code plus one(for sign) minus smallest-real-code) tells how many bits of precision must be carried when performing arithmetic with real numbers.

8.22 VDC EXTENT

<VDC-EXTENT-opcode: 2/8>
<point: first-corner>
<point: second corner>
<IS2>

The parameters first-corner and second-corner are two opposite corners of a rectangle

8.23 CLIP RECTANGLE

<CLIP-RECTANGLE-opcode: 2/9>
<VDC: xmin>
<VDC: xmax>
<VDC: ymin>
<VDC: ymax>
<IS2>

The parameters (xmin,ymin) and (xmax,ymax) are two opposite corners of a rectangle.

8.24 CLIP INDICATOR

<CLIP-INDICATOR-opcode: 2/10>
<enumerated: clip-flag>
<IS2>

<enumerated: clip-flag> = <integer: 0> (on)
| <integer: 1> (off)

8.25 SCALING MODE

<PICTURE-DESCRIPTOR-opcode: 2/1>
<SCALING-MODE-opcode: 3/0>
<enumerated: scaling-mode>
<real: metric-scale-factor>o
<IS2>

<enumerated: scaling-mode> = <integer: 0> (abstract)
| <integer: 1> (metric)

8.26 COLOR SELECTION MODE

<PICTURE_DESCRIPTOR-opcode: 2/1>
<COLOR-SELECTION-MODE-opcode: 3/1>
<enumerated: color-selection-mode>
<IS2>

<enumerated: color-selection-mode> = <integer: 0> (indexed)
| <integer: 1> (direct)

8.27 LINE WIDTH SPECIFICATION MODE

```
<PICTURE_DESCRIPTOR-opcode: 2/1>
<LINEWIDTH-SPECIFICATION-MODE-opcode: 3/2>
<enumerated: specification-mode>
<IS2>

<enumerated: specification-mode> = <integer: 0>  (absolute)
                                 | <integer: 1>  (scaled)
```

8.28 MARKER SIZE SPECIFICATION MODE

```
<PICTURE_DESCRIPTOR-opcode: 2/1>
<MARKER-SPECIFICATION-MODE-opcode: 3/3>
<enumerated: specification-mode>
<IS2>

<enumerated: specification-mode> = <integer: 0>  (absolute)
                                 | <integer: 1>  (scaled)
```

8.29 PERIMETER WIDTH SPECIFICATION MODE

```
<PICTURE_DESCRIPTOR-opcode: 2/1>
<PERIMETER-WIDTH-SPECIFICATION-MODE-opcode: 3/4>
<enumerated: specification-mode>
<IS2>

<enumerated: specification-mode> = <integer: 0>  (absolute)
                                 | <integer: 1>  (scaled)
```

8.30 POLYLINE

```
<POLYLINE-opcode: 3/1>
<point-list>
(<IS1> <point-list>)*
<IS2>
```

8.31 POLYMARKER

```
<POLYMARKER-opcode: 3/0>
<point-list>
(IS1> <point-list>)*
<IS2>
```


8.32 POLYGON

```
<POLYGON-opcode: 3/2>
<point-list>
(<IS1> <point-list>)*
<IS2>
```

8.33 CIRCLE

```
<CIRCLE-opcode: 3/5>
<point: center-of-circle>
<VDC: radius-of-circle>
<IS2>
```

8.34 ARC

```
<ARC-opcode: 3/6>
<point: starting point>
<point: intermediate point>
<point: ending point>
<IS2>
```

8.35 ARC CLOSE

```
<ARC-CLOSE-opcode: 3/7>
<point: starting point>
<point: intermediate point>
<point: ending point>
<enumerated: close-type>
<IS2>

<enumerated: close-type> = <integer: 0>  (pie)
                         | <integer: 1>  (chord)
```

8.36 TEXT

```
<TEXT-or-APPEND-TEXT-opcode: 3/4>
<enumerated: TEXT-final-or-not-final>
<point: starting-point>
<string: text-to-be-displayed>
<IS2>

<enumerated: TEXT-final-or-not-final> = <integer: 0> (final)
                                      | <integer: 1> (not final)
```

The string parameter may contain characters from the VDM G-set or characters from other G-sets, together with the ISO 2022 designating and/or invoking controls to select those other G-sets and to return to the VDM G-set. If the text string contains characters other than from the VDM G-set, prior agreement is required.

Within the string parameter, characters from the VDM G-set are displayed using the character set selected by the most recent CHARACTER SET INDEX element or by the default character set. If the character set selected by CHARACTER SET INDEX (or by default) is a 94-character G-set, the bit combinations 2/0 and 7/15 represent SPACE and DELETE, respectively.

If G-sets other than the VDM G-set are used within the string parameter, the metafile must return to the VDM G-set at or before the end of the string parameter so that subsequent metafile elements can be interpreted correctly.

The following ISO 2022 controls may occur within the string parameter provided there is prior agreement to do so.

(a) In a 7-bit or 8-bit environment:

```
<SO> = 0/14        (SHIFT OUT; invokes the current G1 set)
<SI> = 0/15        (SHIFT IN; invokes the current G0 set)
<LS2> = <ESC> 6/14 (LOCKING SHIFT TWO)
<LS3> = <ESC> 6/15 (LOCKING SHIFT THREE)
```

(b) In an 8-bit environment only:

```
<LS1R> = <ESC> 7/14  (LOCKING SHIFT ONE RIGHT)
<LS2R> = <ESC> 7/13  (LOCKING SHIFT TWO RIGHT)
<LS3R> = <ESC> 7/12  (LOCKING SHIFT THREE RIGHT)
```

(c) Escape sequences to designate character sets as G0, G1, G2, or G3 sets.

8.37 APPEND TEXT

```
<TEXT-or-APPEND-TEXT-opcode: 3/4>
<enumerated: APPEND-TEXT-final-or-not-final>
<string: text-to-be-displayed>
<ISO>

<enumerated: APPEND-TEXT-final-or-not-final>
   = <integer: 2>  (final)
   | <integer: 3>  (not final)
```

See Section 8.36, TEXT, for a description of the parameters for APPEND TEXT.

8.38 CELL ARRAY

```
<CELL-ARRAY-opcode: 3/3>
<P: point>
<Q: point>
<R: point>
<integer: dx>
<integer: dy>
<RGB-list> | <color-index-list>
<IS2>

<RGB-list> = <dx*dy RGB values>

<color-index-list> = <sublist>+

<sublist> = <runlength-coded-sublist>
          | <bit-stream-coded-sublist>

<runlength-coded-sublist> =
                            <runlength-coded-sublist-introducer>
                            <run>+

   <runlength-coded-sublist-introducer> = 2/1

   <run> = <integer: color-index> <integer: number-of-pixels>

<bit-stream-coded-sublist> =
                           <bit-stream-coded-sublist-introducer>
                            <bit-stream>

   <bit-stream-coded-sublist-introducer> = 2/2

                  +-+-+-+-----------+
   <bit-stream> = |X|1|b b b b b b| +
                  +-+-+-+-----------+
```

(Six bits from each byte are deemed to be concatenated into a continuous bit stream bbbbbbbbbbbb... If COLOR INDEX PRECISION is N, the bit stream is grouped into clumps of N bits each. Each such clump represents the color-index of one cell in the cell array.)

8.39 SET ASPECT SOURCE FLAGS

```
<SET-ASPECT-SOURCE-FLAGS-opcode: 5/15>
<aspect-pair>
{<IS1> <aspect-pair>}*
<IS2>

<aspect-pair> = <aspect> <aspect-source>

<aspect> = integer from 0 to 17 specifying the aspect

<aspect-source> = <integer:0> {individual}
                | <integer:1> {bundled}
```

8.40 POLYLINE BUNDLE INDEX

```
<POLYLINE-BUNDLE-INDEX-opcode: 4/4>
<integer: bundle-index>
<IS2>
```

8.41 LINE TYPE

```
<LINE-TYPE-opcode: 4/5>
<index: line-type>
<IS2>

<index: line-type> = <integer: 0>          {solid}
                   | <integer: 1>          {dash}
                   | <integer: 2>          {dot}
                   | <integer: 3>          {dash-dot}
                   | <integer: 4>          {dash-dot-dot}
                   | <integer: negative>   {private line type}
```

8.42 LINE WIDTH

```
<LINE-WIDTH-opcode: 4/6>
<VDC: line-width> | <real: line width scale factor>
<IS2>
```

8.43 LINE COLOR

```
<LINE-COLOR-opcode: 4/7>
<color-specifier>
<IS2>

<color-specifier>
   = <integer: color-index>{if COLOR SELECTION MODE is indexed}
   | <RGB>                 {if COLOR SELECTION MODE is direct}
```


8.44 POLYMARKER BUNDLE INDEX

```
<POLYMARKER-BUNDLE-INDEX-opcode: 4/8>
<integer: polymarker-bundle-index>
<IS2>
```

8.45 MARKER TYPE

```
<MARKER-TYPE-opcode: 4/1>
<index: marker-type>
<IS2>

<index: marker-type> = <integer: 0>          {dot}
                     | <integer: 1>          {plus}
                     | <integer: 2>          {asterisk}
                     | <integer: 3>          {circle}
                     | <integer: 4>          {x}
                     | <integer: negative>{private marker type}
```

8.46 MARKER SIZE

```
<MARKER-SIZE-opcode: 4/2>
<VDC: marker-size> | <real: marker size scale factor>
<IS2>
```

8.47 MARKER COLOR

```
<MARKER-COLOR-opcode: 4/3>
<color-specifier: marker-color>
<IS2>

<color-specifier>
   = <integer: color-index>{if COLOR SELECTION MODE is indexed}
   | <RGB>                 {if COLOR SELECTION MODE is direct}
```

8.48 FILL AREA BUNDLE INDEX

```
<FILL-BUNDLE-INDEX-opcode: 4/8>
<integer: fill-area-bundle-index>
<IS2>
```

8.49 INTERIOR STYLE

```
<INTERIOR-STYLE-opcode: 4/9>
<index: interior-style>
<enumerated: perimeter-visibility>
<ISZ>

<index: interior-style> = <integer: 0>          (hollow)
                        | <integer: 1>          (solid)
                        | <integer: 2>          (pattern)
                        | <integer: 3>          (hatch)
                        | <integer: negative>   (private style)

<enumerated: perimeter-visibility> = <integer: 0>  (off)
                                   | <integer: 1>  (on)
```

8.50 FILL COLOR

```
<FILL-COLOR-opcode: 4/10>
<color-specifier>
<ISZ>

<color-specifier>
  = <integer: color-index>(if COLOR SELECTION MODE is indexed)
  | <RGB>                 (if COLOR SELECTION MODE is direct)
```

8.51 HATCH INDEX

```
<HATCH-INDEX-opcode: 4/11>
<integer: hatch-index>
<ISZ>
```

8.52 PATTERN INDEX

```
<PATTERN-opcode: 4/12>
<integer: pattern-index>
<ISZ>
```


8.53 PATTERN TABLE

```
<PATTERN-TABLE-opcode: 4/15>
<integer: pattern-table-index>
<integer: M>
<integer: N>
<color-specifier> ... (MN of these)
<ISZ>

<color-specifier>
  = <integer:color-index>(if COLOR SELECTION MODE is indexed)
  | <RGB>                (if COLOR SELECTION MODE is direct)
```

8.54 PATTERN REFERENCE POINT

```
<PATTERN-REFERENCE-POINT-opcode: 4/14>
<point: pattern-reference-point>
<ISZ>
```

8.55 PATTERN SIZE

```
<PATTERN-SIZE-opcode: 4/13>
<VDC: delta-x>
<VDC: delta-y>
<ISZ>
```

8.56 PERIMETER TYPE

```
<PERIMETER-TYPE-opcode: 5/0>
<index: perimeter-type>
<ISZ>

<index: perimeter type>
   = <integer: 0>          (solid)
   | <integer: 1>          (dash)
   | <integer: 2>          (dot)
   | <integer: 3>          (dash-dot)
   | <integer: 4>          (dash-dot-dot)
   | <integer: negative>   (private perimeter type)
```

8.57 PERIMETER WIDTH

```
<PERIMETER-WIDTH-opcode: 5/1>
<VDC: perimeter-width> | <real: perimeter-width-scale-factor>
<ISZ>
```

8.58 PERIMETER COLOR

```
<PERIMETER-COLOR-opcode: 5/2>
<color-specifier>
<IS2>

<color-specifier>
  = <integer: color-index>(if COLOR SELECTION MODE is indexed)
  | <RGB>                 (if COLOR SELECTION MODE is direct)
```

8.59 CHARACTER SET INDEX

```
<CHARACTER-SET-INDEX-opcode: 5/13>
<integer: character-set-index>
<IS2>
```

Specifies one of several character sets to be used for subsequent display of graphic text in the TEXT and APPEND TEXT elements.

The mapping between character set indexes and particular character sets is established by the CHARACTER SET LIST element.

8.60 TEXT BUNDLE INDEX

```
<TEXT-BUNDLE-INDEX-opcode: 5/3>
<enumerated: TEXT-BUNDLE-INDEX>
<integer: text-bundle-index>
<IS2>
```

8.61 TEXT FONT INDEX

```
<TEXT-FONT-INDEX-opcode: 5/4>
<integer: font-index>
<IS2>
```

8.62 TEXT PRECISION

```
<TEXT-PRECISION-opcode: 5/5>
<enumerated: TEXT-PRECISION>
<IS2>

<enumerated: text-precision> = <integer: 0>  (string)
                             | <integer: 1>  (character)
                             | <integer: 2>  (stroke)
```


8.63 CHARACTER EXPANSION FACTOR

```
<CHARACTER-EXPANSION-FACTOR-opcode: 5/6>
<real: expansion-factor>
<IS2>
```

8.64 CHARACTER SPACING

```
<CHARACTER-SPACING-opcode: 5/7>
<real: character-spacing>
<IS2>
```

8.65 TEXT COLOR

```
<TEXT-COLOR-opcode: 5/8>
<color-specifier>
<IS2>

<color-specifier>
  = <integer: color-index>(if COLOR SELECTION MODE is indexed)
  | <RGB>                 (if COLOR SELECTION MODE is direct)
```

8.66 CHARACTER HEIGHT

```
<CHARACTER-HEIGHT-opcode: 5/9>
<VDC: character-height>
<IS2>
```

8.67 CHARACTER ORIENTATION

```
<CHARACTER-ORIENTATION-opcode: 5/10>
<VDC: x-component of up vector>
<VDC: y-component of up vector>
<VDC: x-component of base vector>
<VDC: y-component of base vector>
<IS2>
```

8.68 CHARACTER PATH

```
<CHARACTER-PATH-opcode: 5/11>
<enumerated: CHARACTER-PATH>
<IS2>

<enumerated: character-path> = <integer: 0>  (right)
                             | <integer: 1>  (left)
                             | <integer: 2>  (up)
                             | <integer: 3>  (down)
```

8.69 TEXT ALIGNMENT

```
<TEXT-ALIGNMENT-opcode: 5/12>
<enumerated: horizontal-alignment>
<enumerated: vertical-alignment>
<real: continuous-horizontal-alignment> o
<real: continuous-vertical-alignment> o
<ISZ>

<enumerated: horizontal-alignment>
  = <integer: 0>  (left)
  | <integer: 1>  (center)
  | <integer: 2>  (right)
  | <integer: 3>  (normal horizontal)
  | <integer: 4>  (continuous horizontal)

<enumerated: vertical-alignment>
  = <integer: 0>  (top)
  | <integer: 1>  (cap)
  | <integer: 2>  (half)
  | <integer: 3>  (base)
  | <integer: 4>  (bottom)
  | <integer: 5>  (normal vertical)
  | <integer: 6>  (continuous vertical)
```

8.70 COLOR TABLE

```
<COLOR-TABLE-opcode: 5/14>
<integer: starting-point>
<color-list>
<ISZ>

<color-list> = <RGB>+
```

8.71 VDM ESCAPE

```
<VDM-ESCAPE-opcode: 7/13>
<integer: identifier>
<string: opcode-and-parameters-of-private-command>
<ISZ>
```

8.72 MESSAGE

```
<MESSAGE-opcode: 2/12>
<enumerated: action-flag>
<string: message-to-be-displayed>
<ISZ>

<enumerated: flag> = <integer: 0>  (yes)
                   | <integer: 1>  (no)
```


8.73 APPLICATION DATA

```
<APPLICATION-DATA-opcode: 2/13>
<string: identifier>
<string: application-data>
<ISZ>
```

9. Character Encoding Defaults

VDM PRECISIONS for the character-coded graphics encoding:

Non-VDC Reals: exponent, 4 bits; integral mantissa, 15 bits
Non-VDC Integers: 16 bits
Color direct: 6 bits red, 6 bits green, 6 bits blue
Index: 6 bits
Color index: 6 bits
Real VDC: exponent, 10 bits; integral mantissa, 15 bits
Integer VDC: 17 bits

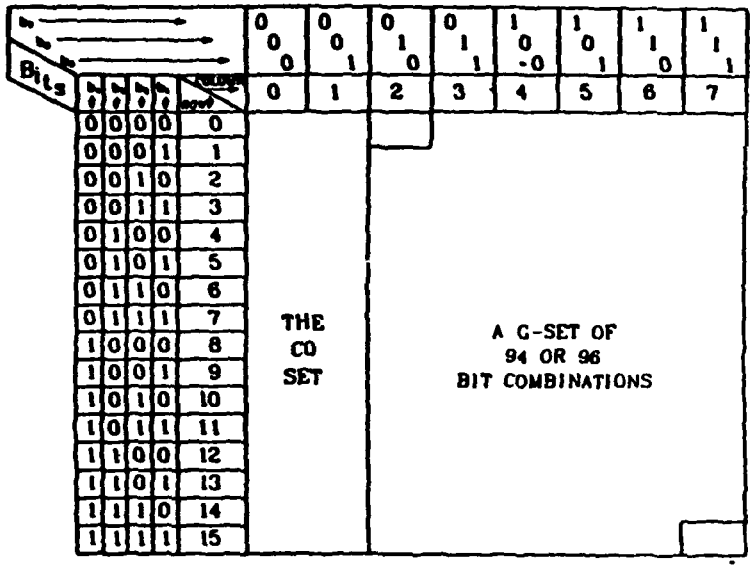


Figure 1: The 7-bit code table.

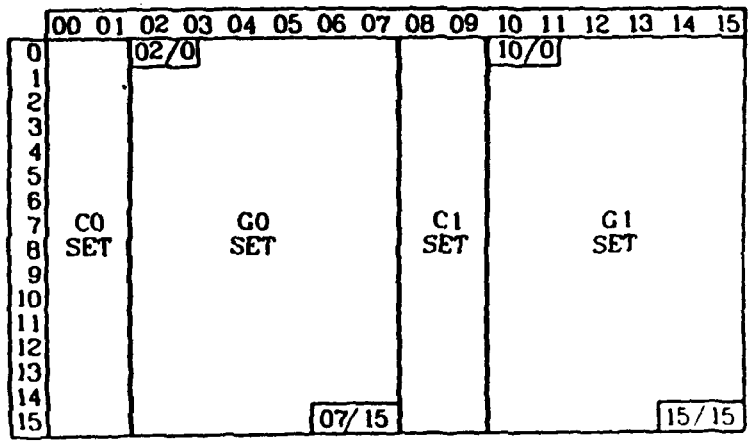


Figure 2: The 8-bit code table.

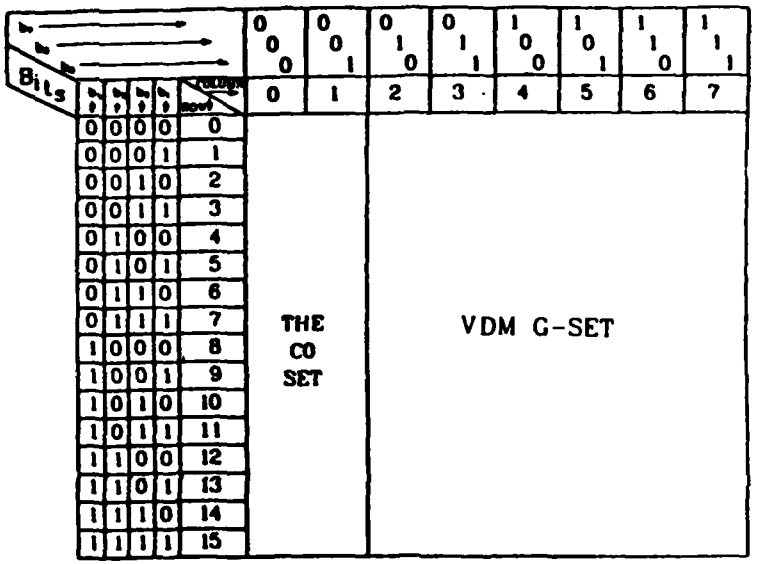
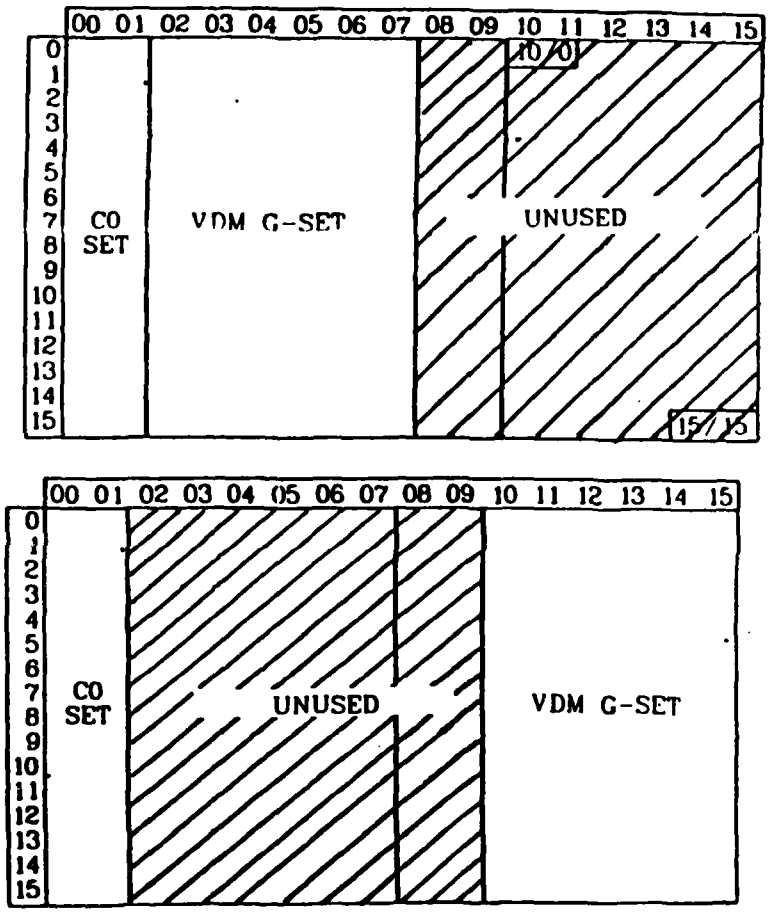


Figure 3: The 7-bit VDM code table.



APPENDIX A

CHARACTER CODED GRAPHICS ENCODING-DEPENDENT FORMAL GRAMMER

This Appendix is not a part of American National Standard X3.nnn-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

In this encoding scheme, a metafile element always has the form

```
<metafile element>          ::= <opcode>
                                <parameter character>*
                                <ISI>
```

One of the parameter characters can be <ISI>, which is used to separate parameter lists. If it occurs in the parameter characters of POLYLINE, POLYGON, or POLYMARKER, it indicates a repetition of the same element. Details describing these cases can be found in Section 7.4.

A VDM *opcode* is encoded as a one-byte character or as a one-byte character followed by an enumerated type. In the second case, two or more VDM elements share the same primary opcode character, and a secondary opcode indicates the particular *element*. This scheme is used for the Metafile Descriptor elements, the Picture Descriptor elements, the TEXT-or-APPEND-TEXT element, and the BEGIN-METAFILE-or-END-METAFILE element.

The enumerated types are represented by integers. The specific integer values for each enumerated type are listed in Section 8 in the format

```
    <integer: 0>   (first enumerated element)
  | <integer: 1>   (second enumerated element)
  | <integer: 2>   (third enumerated element)
    .
    .
    .
```

The other terminal symbols are described in detail in Section 8. A reference to the relevant sections is given here.

```
<integer>                   ::= <more significant bits>*
                                <least significant bits>
```

where the least significant bits contain the sign bit. See Section 7.1.

```
<real>                      ::= <integral mantissa>
                                <exponent>(e)
```

<integral mantissa> ::= <integer>

<exponent> ::= <integer>

where the least significant bits of the integral mantissa indicate whether an exponent follows.

<coordinate> ::= <integer> <integer>
| <real> <real>

where a coordinate list is encoded such that the first coordinate is absolute within VDC space and each of the following coordinates is relative to its previous one. See Section 7.3.

<vdc value> ::= <integer>
| <real>

<string> ::= <character>* <IS1>
| <character>* <IS2>

<character> ::= a character from the
7-bit or 8-bit code tables,
as discussed in Section 2.1.

<character substitution> ::= <string> as described in
Section 6.

<color index> ::= <integer>

<red green blue> ::= <bits for red>
<bits for green>
<bits for blue>

The packing of these bits is described in Section 7.5.

<integer prec value> ::= <integer> See Section 8.4

<real prec value> ::= <integer>
<integer>
See Section 8.5

<index prec value> ::= <integer> See Section 8.6

<color prec value> ::= <integer> See Section 8.7

<col index prec value> ::= <integer> See Section 8.8

<integer vdc prec value> ::= <integer> See Section 8.20

<real vdc prec value> ::= <integer> See Section 8.21

<color list>	::= <run length coded sublist> | <bit stream coded sublist>
<run length coded sublist>	::= <run length opcode> <run>*
<run>	::= <integer: color index> <integer: number of cells>
<bit stream coded sublist>	::= <bit stream opcode> <bit stream>
<bit stream>	::= bits of concatenated bytes. See Section 7.6
<escape data list>	::= <string> See Section 8.71

Part 3 VIRTUAL DEVICE METAFILE BINARY ENCODING

0. Introduction

The primary purpose of a binary encoding is to minimize VDM generator and interpreter overhead. Binary encoded VDM elements are easy to format, read and scan.

This section defines a standard binary encoding suitable for the Virtual Device Metafile. Other binary encodings are possible; however, this section defines the standard binary encoding. Implementations of this encoding must conform exactly to be standard. This encoding combines the best features from current practice and the requirements of GKS and VDM. Only the format for the various VDM elements is defined. How this information is transmitted or recorded is left to other standards.

dpANS VDM

1. Virtual Device Binary Metafile Structure

The Virtual Device Binary Metafile (VDBM) is a bit stream containing graphical and control information in a standard format. There are two components of a VDBM: the Metafile Descriptor (MD), which provides information on how to correctly interpret the VDBM, and the VDM body. The VDM body is partitioned into pictures that are units of pictorial information; logically, these are single images. Pictures are, in turn, comprised of graphical elements for describing picture components, and attribute elements for describing the appearance of the graphical elements. Control elements provide direction for the interpretation processor.

The elements in the VDBM are represented as variable length data structures, each consisting of an optode designating the particular VDM element, the length of the parameter list (in bytes), and the list of parameters.

The first VDM command will normally be at the beginning of the file. The following is a typical structure of a VDBM.

BEGIN MF.	MD	<picture 1> ... <picture i> ...	END MF.

2. Pictures

VDM data is partitioned into pictures. All pictures are mutually independent. A picture consists of a BEGIN PICTURE command, control, graphical, and attribute elements, and an END PICTURE command.

```
-------------------------------------------------------------------
| <BEGIN PIC i> <ELEMENT> <ELEMENT>...<ELEMENT> <END PIC i>  |
-------------------------------------------------------------------
```

To guarantee that each picture will be output correctly, the VDM interpreter must reset all appropriate elements to their defaults at the BEGIN PICTURE command. See Part I Appendix B for Interpreter Guidelines.

3. Binary Encoding Features

This binary encoding has the following features:

- Each VDM element is represented by a VDBM command. A command consists of an opcode plus associated parameter(s).
- A null/no-op command is available. Since every command is valid data, this command may be used to reach machine-dependent record boundaries.
- Every command begins on a Metafile word boundary. Commands are padded with a zero byte when necessary to finish on a word boundary.
- All opcodes have an associated parameter length value. The length is specified as a byte count.
- Commands are encoded as 16-bit words.
- Coordinates are restricted to start on word boundaries.
- Other data such as indexes, color, and characters are encoded as one or more bytes.
- In each word, or unit within a word, the bit with the highest number is the most significant bit. Likewise, when commands are sequentially accessed, the least significant word (command or operand) follows the most significant.
- Future growth (new data types and graphical elements) is allowed.
- Unnecessary data for short elements is avoided.
- Commands and data are easy to format to minimize interpreter/generator overhead.
- Fast scanning of the VDBM without excessive interpretation is possible.
- Floating-point numbers are encoded using IEEE floating-point representation.
- Parameter lengths are determined by the appropiately specified precisions.

4. Data Structure

The smallest addressable unit of data used in this encoding is an 8-bit byte. A word consists of two bytes (16 bits). (This VDBM word size should not be confused with a physical machine word, which may be different.) To insure portability of the VDBM, the maximum precision for reals is restricted to four words (64 bits), and the maximum precision for index, integer, color index and color parameters is restricted to two words (32 bits).

VDM elements are represented with a command of varying length. The structure of the commands is given below. A 'short-command' consists of a word partitioned into three fields indicating the element-class, element-id, and parameter-list-length. The 'long-command' form is similar to the 'short-command' form except that an additional word is used for the parameter-list-length. In addition, the 'long-command' form allows data to be partitioned. This is accomplished by reserving the high-order bit of the second word to indicate if this is the last or a continued data partition. If the next data partition is specified, the parameter-count word and associated parameters will follow the last parameter of the previous partition. More information on partitioning is given in Appendix A, Formal Grammar.

Word 1->	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Element-class				Element-id							Parameter-list-length				

If Parameter-list-length=11111 (31), an additional word follows with the desired value.

Word 2->	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	P	Parameter-list-length														

Word 1:

bits 15-12 = element-class (allows up to 16 classes)

bits 11-5 = element-id (allows up to 128 VDBM elements within each class)

bits 4-0 = parameter-list-length (the number of bytes that follow for this command (<= 30))

Word 2:

bit 15 = partition flag (0 for last partition, 1 for next partition)
bits 14-0 = parameter-list-length (the number of bytes that follow for this command (<= 32767))

The list of parameter values *follows* the parameter-list-length for either the 'long' or 'short' command form. The number of values is determined from the parameter-list-length, and the type and precision of the operands. These parameter-values have the format illustrated in Section 5. The parameter-type for coordinates is indicated in the MD. For noncoordinate parameters, the parameter-type is as specified in Part 1 Section 5. If the parameter-type is encoding dependent, its VDBM code is specified in the coding tables in Sections 8 through 12. Unless otherwise stated, the order of the parameters is as listed in Part 1 Section 5.

5. Primitive Data Forms

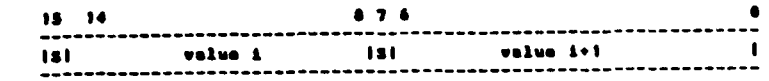
The binary encoding of the VDM uses a handful of primitive data forms to represent the various abstract data types of the VDM. These primitive data forms are defined in this section.

The forms defined are Signed Integer (SI), Unsigned Integer (UI), Character (C), and Floating Point (FP). Each of these primitive forms (except characters) can be used in a number of precisions. The definitions below show the allowed precisions for each primitive data form.

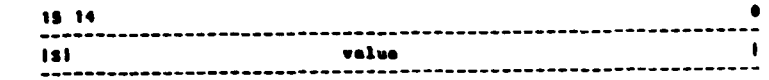
The definitions are in terms of "metafile words" which are 16-bit units.

Signed Integer (Symbol = SI) (two's complement):

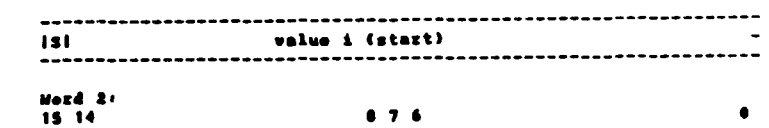
At 8-bit precision.



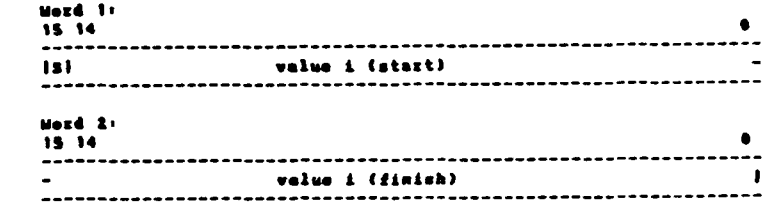
At 16-bit precision.



At 24-bit precision (each value straddles 2 metafile words):

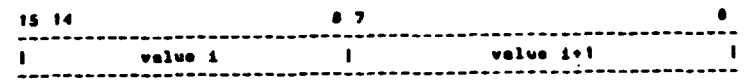


At 32-bit precision (each value fills 2 metafile words).

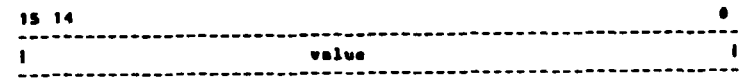


Unsigned Integer (Symbol =UI):

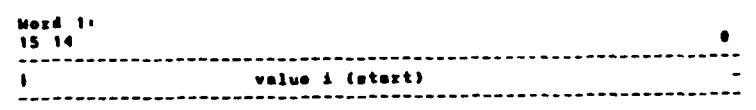
At 8-bit precision.



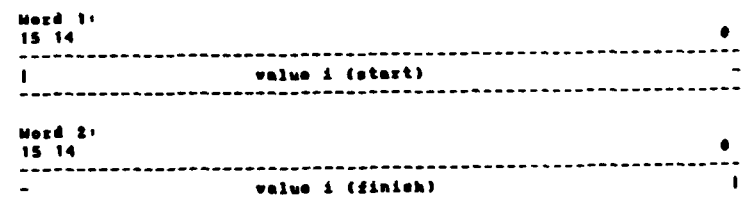
At 16-bit precision.



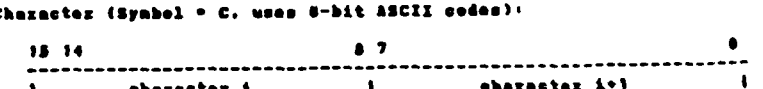
At 24-bit precision (each value straddles 2 metafile words).



At 32-bit precision (each value fills 2 metafile words).

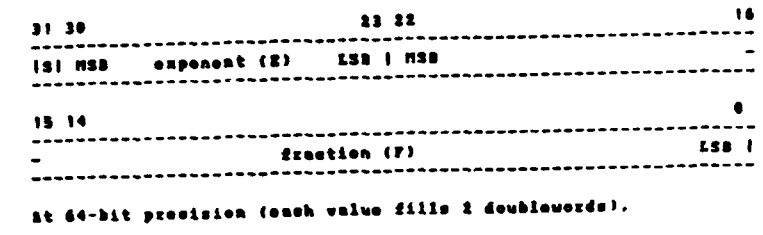


Character (Symbol = C, uses 8-bit ASCII codes):

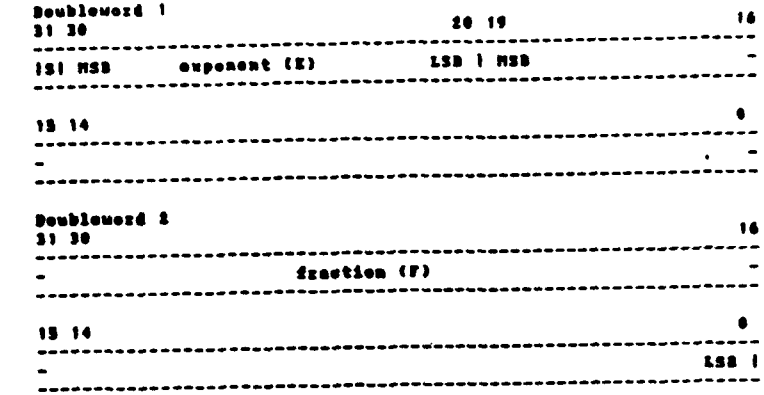


Floating Point (Symbol = FP, uses IEEE Format), shown in terms of "metafile doublewords", 32-bit:

At 32-bit precision (each value fills 1 doubleword).



At 64-bit precision (each value fills 2 doublewords).



MSB Most Significant Bit
LSB Least Suignificant Bit
S 1 bit sign
E Exponent (biased appropiately for precision)
F Fraction (mantissa)
Value $(-1)^{sign} * 1^{sign} * (E-B)^{2} * (1.F)$ where B (bias) is 127 or 1023

Note: For degenerate cases, see the IEEE floating point standard.

6. Parameter Type Representation

The following table shows, for each of the abstract parameter types, how it is represented in the binary metafile in terms of primitive data forms. First, the symbol for the abstract parameter type is given, as it is used in Section 5 of the VDM functional specification. Then it is stated how the parameter type is constructed in terms of the primitive data forms, at the appropriate precisions. The precisions are those defined in Section 5 of the VDM functional specification.

Next is given the symbol for the number of bytes required to represent one instance (occurrence) of the given parameter, at the given precision , and the formula for computing the number. Finally, the symbol is given for the range of values which the parameter can assume, followed by the numerical values which the parameter can assume, followed by the numerical values which define the range. These latter two symbols are used extensively in the code tables in the following sections.

+IR denotes the positive integers. Repeats are denoted by: mI(i.e. m occurances of I), I^n, etc. Combinations are used: IR, 2I, IX^n, etc. Lists are used: I, I, I/R, etc.

Table 1 Parameter-Type Representations

Abstract Parameter Type Symbol	Parameter Construction from Primitive Forms	Bytes/Parameter Symbol and Value	Parameter Range Symbol and Value
CI	UI direlor index precision (cip)	BCI [= cip/8] see note 1 & 2	CIR [0 to (2**cip - 1)]
CD	3UI ddirect color precision (dcp)	BCD [= 3*dcp/8]	CDR [0 to (2**dcp - 1)]
IX,E	SI d index precision (ixp)	BIX [= ixp/8] see note 3	IXR [-2**(ixp-1) to 2**(ixp-1)-1]
I	SI d integer precision (ip)	BI [= ip/8]	IR [-2**(ip-1) to 2**(ip-1)-1]
R	FP d real precision (rp)	BFP [=FP/8] [see note 4]	FPR [see note 5]
S	UI,nC	BS+1 [= n+1]	SR [see note 6]
VDC	SI d VDC integer precision (vip)	BVDC [= vip/8]	VDCR [-2**(vip-1) to 2**(vip-1)-1]
	or	or	or
	FP d VDC real precision (vip)	BVDC [=vip/8] [see note 4]	VDCR [see notes 1,5,7&8]
P	(VDC,VDC)	BP [= 2*BVDC]	VDCR [see notes 1,5,7&8]

The following is not an abstract parameter type, but a symbol definition that is convenient shorthand:

CO	CI	BCO [= BCI]	COR [see note 9] [= CIR]
	or	or	or
	CD	BCO [= BCD]	COR [= CDR]

Note 1: For parameters that are composed of multiple identical components (e.g., direct color, CD, and Point, P) the range value represents the range of a single component.

Note 2: Direct color is abstractly a real in the range [0,1). This is normalized onto the unsigned range in the table above.

Note 3: Abstract parameter type Enumeration, E, is treated in the binding identically to abstract type Index, IX. The binding uses the IX symbols to handle both.

Note 4: FP is the sum of the exponent and fraction precision given in the real precision element. In the VDM body, the Real VDC Precision element may cause FPR to be updated.

Note 5: FPR, VDCR is computed following the IEEE standard. (See Section 5 Floating Point Data Form)

Note 6: The range for parameter type S is not applicable. The range for character data is not applicable. An one byte unsigned integer is for the number of characters. Some character sets may require additional data bytes. See CHARACTER SET LIST element.

Note 7: The abstract parameter type VDC, a single VDC value, is either a real or an integer, depending upon the declaration of the metafile descriptor function VDC TYPE. Subsequent tables use a single set of symbols, VDC, BVDC, and VDCR, recognizing that they are computed differently depending upon VDC type.

Note 8: The abstract parameter type VDC is a single VDC value. A Point, P, is an ordered pair of VDC.

Note 9: The parameter-type symbol CO does not correspond to an abstract parameter type from the VDM functional specification. Rather it is a convenient shorthand for "color", which is either direct color (CD) or indexed color (CI), depending upon the value of the VDM element COLOR SELECTION MODE. The associated bytes per parameter and range symbols, BCO and COR, are thus either BCI and CIR or BCD and CDR, respectively, depending upon COLOR SELECTION MODE.

7. Coding

The code tables are in two parts. The first part specifies the VDM element, element-code, parameter type, parameter-list-length, parameter range, and default values. The parameter-list-length is given in bytes, which in some cases is constant and in other cases is variable. Since all VDSM elements must begin on a word (16 bit) boundary, some commands must be padded with an 8-bit byte. The parameter-list-length includes this extra byte. Additional information on defaults is given in Part 1 Section 6. The second part of each table provides additional element coding details. Functional meanings and parameter descriptions are given in Part 1 Section 5.

Table 2 gives the coding for the classes defined in Part 1 Section 4.

Table 2: Classes

Class	VDSM Item
0	null
1	metafile descriptor
2	control elements
3	picture descripter elements
4	graphical elements
5	attribute elements
6	escape elements
7	external elements
8-15	reserved for future standardisation

8. Metafile Descriptor

The Metafile Descriptor is associated with a VDM file. It provides information on format, identification, and defaults to be used in interpreting the VDBM. It must be read and interpreted prior to reading the body of the VDBM. Table 3 gives the items that may be found in the MD.

Table 3: Metafile Descriptor Coding

VDBM Element (Class Code = 1)	Element Code	Parameter Type	Parameter List Length	Parameter Range	Default
VDM VERSION	1	I	BI	+IB(0...n)	1
VDM DESCRIPTION	2	S	BS	SB	null
VDC TYPE	3	E	BIX	1,2	1
INTEGER PRECISION	4	I	BI	8,16,24,32	16
REAL PRECISION	5	2I	2BI	(8,24), (12,52)	8,24
INDEX PRECISION	6	I	BI	8,16,24,32	8
COLOR PRECISION	7	I	BI	"	8
COLOR INDEX PRECISION	8	I	BI	"	8
VDM ELEMENT LIST	9	InE	InBIX	+IXB	n/a
VDM DEFAULTS REPLACEMENT	10	VDBM	variable commands	VDBM	null commands
CHARACTER SET LIST	11	n(E,S)	n(BIX+BS)	(+IXB,SB)	1,ASCII
FONT LIST	12	n(IX,S)	n(BIX+BS)	(+IXB,SB)	1,any
MAXIMUM COLOR INDEX	13	CI	BCI	CIB	0

Element Code	Parameter Description
3	P1: VDC TYPE: 0 =integer, 1 = real.
5	P1: is field width for exponent. P2: is field width for fraction. For double precision, P1 = 12, and P2 = 52.
9	List of VDBM elements in this VDBM. This is enumeration type in Part 1 Section 5, but VDBM uses 2IX for each command. The first IX is the class code; the second IX is the element code.
10	Sublist of VDBM elements with default parameter values desired. The structure and format is identical to appropiate VDBM element(s).
11	P1: SET TYPE: 0=94 character, 1=96 character, 2=2-byte, 3=8-bit code, 4=private code P2: designation sequence tail. See Part 1 Section 5.
12	P1: Index 1=ASCII P2:any font which can represent ASCII.

9. Control Elements

Table 4: Control Element Coding

VDBM Element (Class Code = 2)	Element Code	Parameter Type	Parameter List Length	Parameter Range	Default
no-op	0	S	nC	na	na
BEGIN METAFILE	1	S	SS	SR	null
END METAFILE	2	na	0	na	na
BEGIN PICTURE	3	E,CO,S	2IX+2CO	(0,1), COR,SR	null
BEGIN PICTURE BODY	4	na	0	na	na
END PICTURE	5	na	0	na	na
INTEGER VDC PREC	6	I	2I	16,32	16
REAL VDC PREC	7	I,I	2*2I	(8,24), (11,52)	8,24
VDC EXTENT	8	2P	2*3P	VDCR	see below
CLIP RECTANGLE	9	4VDC	4*3VDC	VDCR	see below
CLIP INDICATOR	10	IX	2IX	(0,1)	0
LOCAL BACKGROUND COLOR	11	E,CO	2IX+2CO	(0,1),COR	0, (1,1,1)

Element Code	Parameter Description
0	The no-op command can be used to pad the VDBM data stream. If the parameter count indicates a string is available, it is read as 8-bit character bytes and can be used as a comment.
1	This is the first element of a VDBM. For this encoding, it is read as a 16-bit word. The next command must indicate the beginning of the MD.
2	This is the last command in a VDBM. No other commands are allowed between END METAFILE and the next (if any) BEGIN METAFILE.
3	Defaults as specified by the VDM DEFAULTS REPLACEMENT command are used with the BEGIN PICTURE command. P1: View Surface Color flag: 0 = device dependent, 1 = use P2. P2: View Surface Color. P3: arbitrary character string.

3,5	All VDSM commands except BEGIN METAFILE, END METAFILE and MD may appear between BEGIN PICTURE and END PICTURE commands.
4	This element must follow the BEGIN PICTURE element.
6	This value resets the integer precision. Legal values are 8, 16, 24, and 32.
7	This element resets the real precision initially defined by the MD. P1: exponent size in bits, legal values are 8 and 18. P2: fraction size in bits, legal values are 24 and 52.
8,9	Default is (0..0.), (1.,1.) if VDC TYPE is Real, or (0,0), (32767,32767) if VDC TYPE is Integer.
10	0 = OFF, 1 = ON
11	P1: 0=off, 1=on

Table 5: Picture Descriptor Element Coding

VDSM Element (Class Code = 3)	Element Code	Parameter Type	Parameter List Length	Parameter Range	Default
SCALING MODE	1	E,R	BIX+BFP	(0,1),FPR	0,1.
COLOR SPECIFICATION MODE	2	E	BIX	(0,1)	0
LINE WIDTH SPECIFICATION MODE	3	E	BIX	(0,1)	1
MARKER SIZE SPECIFICATION MODE	4	E	BIX	(0,1)	1
PERIMETER WIDTH SPECIFICATION MODE	5	E	BIX	(0,1)	1

Element Code	Parameter Description
1	P1: 0=abstract, 1=metric P2: metric scale factor
2	0=indexed, 1=direct
3,4,5	0=absolute, 1=scaled

10. Graphical Elements

Table 6: Graphical Element Coding

VDSM Element (Class Code = 4)	Element Code	Parameter Type	Parameter List	Parameter Range	Default
POLYLINE	1	nP	nBP	VDCR	na
POLYMARKER	2	nP	nBP	VDCR	na
POLYGON	3	nP	nBP	VDCR	na
CIRCLE	4	P,VDC	BP+BVDC	VDCR	na
ARC	5	3P	3BP	VDCR	na
ARC CLOSE	6	3P,E	3BP+BIX	VDCR, (0 or 1)	na
CELL ARRAY	7	3P,2I, mnCO	3BP+2BI+ mnBCO	VDCR,+IR, COR	na
TEXT	8	P,E,S	BP+BIX+BS	VDCR, (0or1),SR	na
APPEND TEXT	9	E,S	BIX+BS	(0or1),SR	na

Element Code	Parameter Description
6	P2: Close type: 0=pie, 1=chord.
8,9	P1: 0=final, 1=not final.

11. Attribute Elements

Table 7: Attribute Element Coding

VDM Element (Class Code = 5)	Element Code	Parameter Type	Parameter List Length	Parameter Range	Default
POLYLINE BUNDLE INDEX	1	IX	BIX	+IXR	1
LINE TYPE	2	IX	BIX	IXR	0
LINE WIDTH	3	VDC or R	BVDC or BFP	+VDCR or +FPR	(VDCR/1000) or 1.
LINE COLOR	4	CO	BCO	COR	see below
POLYMARKER BUNDLE INDEX	5	IX	BIX	+IXR	1
MARKER TYPE	6	IX	BIX	IXR	2
MARKER SIZE	7	VDC or R	BVDC or BFP	+VDCR or +FPR	(VDCR/1000) or 1.
MARKER COLOR	8	CO	BCO	COR	see below
TEXT BUNDLE INDEX	9	IX	BIX	+IXR	1
TEXT FONT INDEX	10	IX	BIX	+IXR	1
TEXT PRECISION	11	E	BIX	1...3	1
CHARACTER EXPANSION FACTOR	12	R	BFP	FPR	1.
CHARACTER SPACING	13	R	BFP	FPR	0.
TEXT COLOR	14	CO	BCO	COR	see below
CHARACTER HEIGHT	15	VDC	BVDC	VDCR	VDCR/100
CHARACTER ORIENTATION	16	4VDC	4BVDC	VDCR	(0,1), (1,0)
CHARACTER PATH	17	E	BIX	0...3	0
TEXT ALIGNMENT	18	IX,IX, R,R	2BIX+ 2BFP	0...4, 0...6, FPR,FPR	0,0, 0.,0.
CHARACTER SET INDEX	19	IX	BIX	+IR	1
FILL AREA BUNDLE INDEX	20	IX	BIX	+IXR	1
INTERIOR STYLE	21	IX,E	2BIX	1...4, (0 or 1)	0 or 1
PATTERN INDEX	22	IX	BIX	+IR	1
HATCH INDEX	23	IX	BIX	+IR	1
FILL COLOR	24	CO	BCO	COR	see below
PATTERN SIZE	25	2VDC	2BVDC	VDCR,VDCR	1/25VDCR

COLOR TABLE	26	CI,nCD	BCI+nBCD	CIR,CDR	see below
PATTERN TABLE	27	IX,IX, nnCO	BIX+2BI +nnBCO	+IXR,+IR, COR	1,(1,1), 0
PATTERN REFERENCE POINT	28	P	BP	VDCR	0,0
SET ASPECT SOURCE FLAGS	29	n(IX,E)	n(2BIX)	(0,17), (0,1)	0,0
PERIMETER TYPE	30	IX	BIX	+IXR	0
PERIMETER WIDTH	31	VDC or R	BVDC or BFP	+VDCR or BFPR	same as linewidth
PERIMETER COLOR	32	CO	BCO	COR	see below

Element Code	Perimeter Description
2,30	0=solid, 1=dash, 2=dot, 3=dash-dot, 4=dash-dot-dot, 0 > implementation dependent
4,8,14, 24,32,	for direct, a 3-tuple of red, green, blue integer values of the specified by the set color precision given. Default is the device dependent foreground color.
6	0=dot, 1=plus, 2=asterisk, 3=circle, 4=x, 0 > implementation dependent.
10	The meaning of the index is specified by MD element 12.
11	text precision: 0=string, 1=character, 2=stroke.
17	P1: 0=right, 1=left, 2=up, 3=down.
18	P1: 0=normal, 1=left, 2=center, 3=right, 4=continuous. P2: 0=normal, 1=top, 2=cap, c=half, 4=base, 5=bottom, 6=continuous.
21	P1: interior style: 0=hollow, 1=solid, 2=hatch, 3=pattern. P2: perimeter visibility: 0=off, 1=on.
27	P1: index P2: array bounds P3: 0 = solid
29	P1: aspect index P2: ASF value: 0 = individual, 1 = bundled

12. Escape, External Elements

Table 8: Escape, External Element Coding

VDM Element	Element Code	Parameter Type	Parameter List Length	Parameter Range	Default
(Class Code = 6)					
VDM ESCAPE	1	I,S	IXR,BS	IXR+SR	na
(Class Code = 7)					
MESSAGE	1	E,S	BIX+BS	0or1,SR	0/na
APPLICATION DATA	2	X,S	2BS	SR	na

Class Code	Element Code	Parameter Description
6	1	P1: identifier string P2: data
7	1	P1: Action required: 0=no, 1=yes
7	2	P1: identifier string P2: data

13. Conformance

A metafile conforms to this VDM if it meets the following requirements:

- Each metafile element described in this section is coded in the manner described

- Private (nonstandard) metafile elements are all coded using the VDM ESCAPE metafile element. Opcodes reserved for future standardization are not used to code private (nonstanard) metafile elements.

- Private (nonstanard) values of index parameters are all coded using negative integers. In coding index parameters, a metafile shall not use nonnegative integers to represent private values of index parameters.

A conforming metafile may include, within the string parameters of TEXT and APPEND TEXT metafile elements, the ISO 2022 controls for designating and invoking G-sets. This is an alternative way, in addition to CHARACTER SET INDEX, by which character sets for displaying text text strings may be selected. However, the use of ISO 2022 controls within text strings is implementation dependent. Metafile interpreters are not required to respond correctly to the ISO 2022 controls for designating and invoking G-sets when these controls occur within a TEXT or APPEND TEXT command.

14. Binary Encoding Defaults

VDM PRECISIONS for the binary encoded metafile:

Word Size: 16 bits; BYTE: 8 bits
Non-VDC Reals: exponent 8 bits, Mantissa 24 bits
Non-VDC Integers: 16 bit, 2's complement
Color direct: 3 BYTES
Color index: 1 BYTE
Index: 1 BYTE
Real VDC: exponent, 8 bits; mantissa, 24 bits
Integer VDC: 16-bit 2's complement

APPENDIX A

BINARY ENCODING-DEPENDENT FORMAL GRAMMAR

This Appendix is not a part of American National Standard X3.nnn-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

The VDM opcodes are encoded as two integers specifying the element-class and the element-id. The element classes are listed in Table 2, and the element-ids are listed in Tables 3 through 8. For example,

<VDM VERSION> ::= 1 1 <operand list length>

<VDM DESCRIPTION> ::= 1 2 <operand list length>

<operand list length> ::= <integer> (encoded as described in Section 4)

The enumerated types are 8-bit or 16-bit integers. Tables 3 through 8 specify which integers are assigned to operands of type "E".

The other terminal symbols are described in detail in Section 5. A reference to the relevant tables is given here:

<integer> ::= two's complement integer. See Section 5

<real> ::= floating-point number (IEEE Standard). See Section 5

<coordinate> ::= <integer>(2) | <real>(2)

<vdc value> ::= <integer> | <real>

<string> ::= <length> <character>*

<length> ::= <integer> (8-bit unsigned)

<character> ::= 8-bit ASCII character, or a number of characters depending on the character set. See Table 1

<character substitution> is not used in this encoding.

<color index> ::= <integer>

<red green blue> ::= three consecutive numbers, encoded as stated in Table 1

The operands:

<index prec value>,
<integer prec value>,
<real prec value>,
<color prec value>,
<col index prec value>
<integer vdc prec value>,
<real vdc prec value>

are described in Table 3.

are described in Table 4.

<color list> is described in Table 7.

The operand:

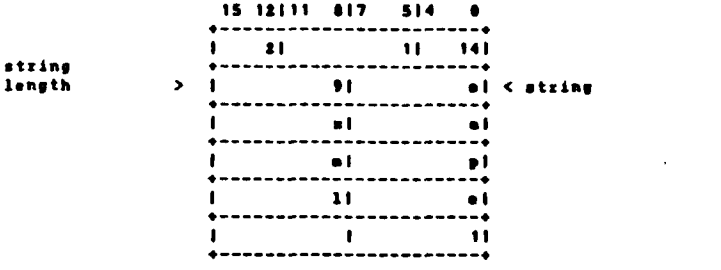
<escape data list> is described in Table 8.

BINARY METAFILE ELEMENT ENCODING EXAMPLES

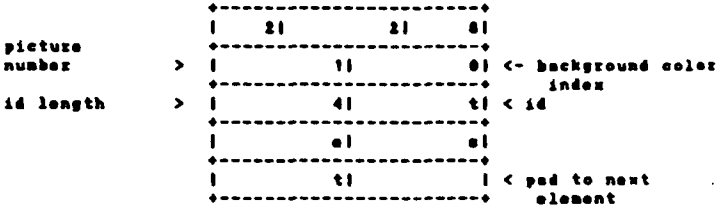
This Appendix is not a part of American National Standard X3.xxx-198x X3.xxx-198x, Draft Proposed American National Standard for the Virtual Device Metafile, but is included for information purposes only.

The following simple examples illustrate the use of the VDBM. All precisions use the default values.

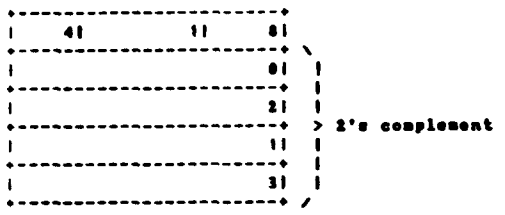
Example 1: Begin Metafile "example 1"



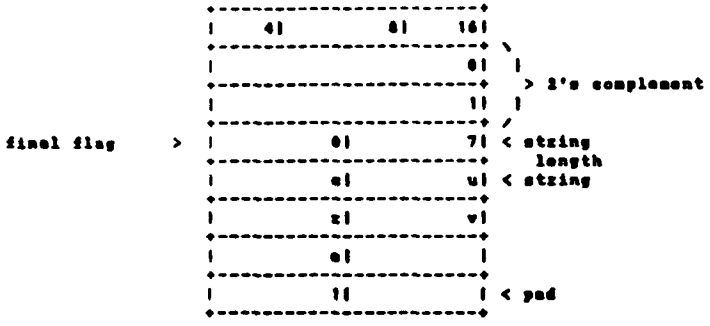
Example 2: Begin Picture 1 "test"



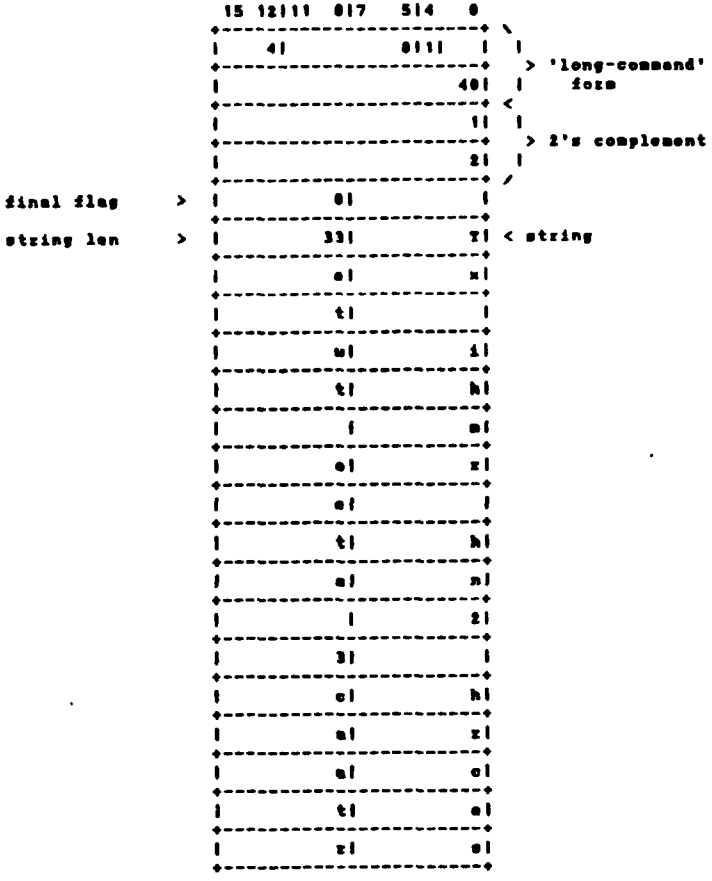
Example 3: Polyline 0,2 to 1,3



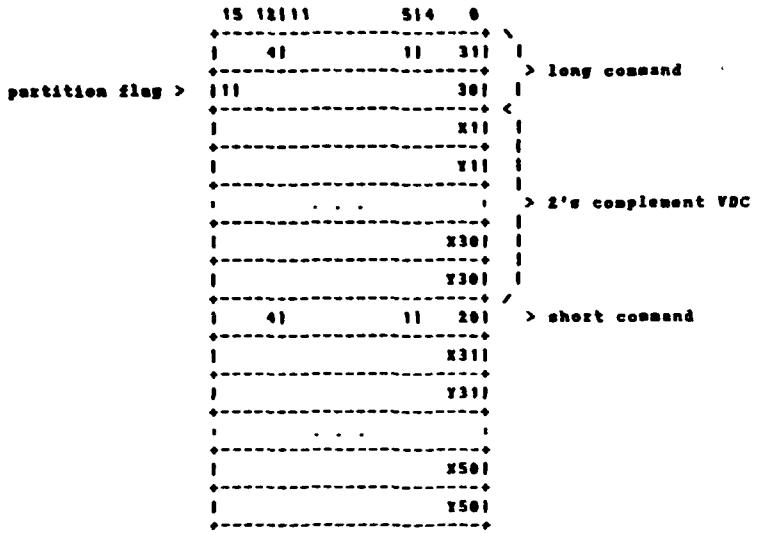
Example 4: Text "Curve 1" at 0,1



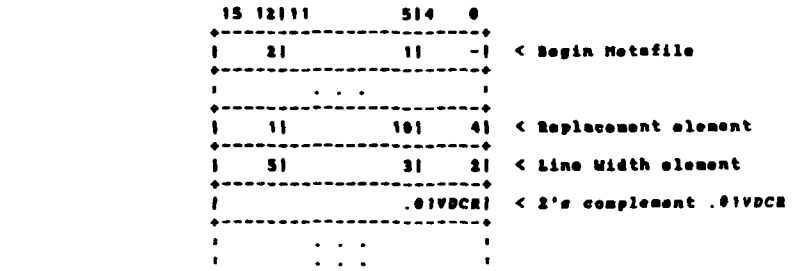
Example 5: Text "Text with more than 23 characters" at 1,2



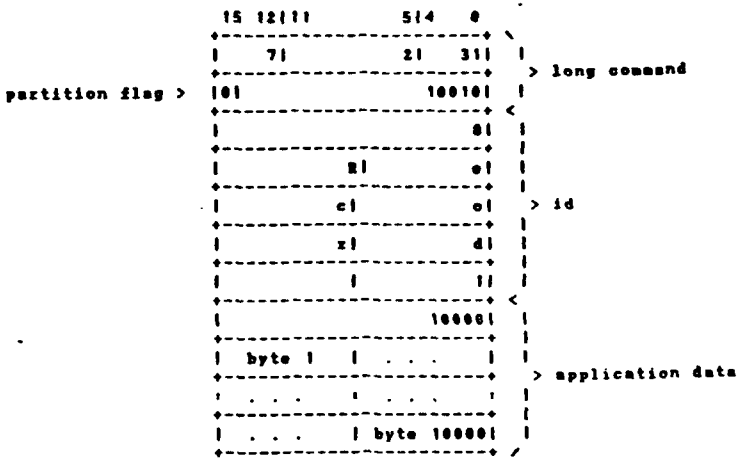
Example 6 Partitioned Polyline with 50 points



Example 7 VDM Default Replacement for Line Width to .01*VDCR



Example 8 Application Data "Record 1" with 10000 bytes of data



APPENDIX D

ANSI Document X3H3/84-97
Summary of comments received during first
public review of dpANS VDM (X3.122-198x)

Comment Number: 01-001, 09-001, 09-002

Comment Topic: Graphical Output

Comment: Add mechanism to specify "holes" in polygons and separate polygon edge visibility control

Action: Accepted

Response: A new element POLYGON SET has been added to the VDM. Part 1 Section 5.5.5 describes the new element POLYGON SET. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 01-002

Comment Topic: Graphical Output

Comment: Add a means of specifying such that the center point and the radius are given explicitly

Action: Accepted

Response: New elements CIRCULAR ARC CENTER and CIRCULAR ARC CENTER CLOSE have been added to the VDM. Part 1 Section 5.5.10 and Section 5.5.11 describe the new elements CIRCULAR ARC CENTER and CIRCULAR ARC CENTER CLOSE. The names of the elements ARC and ARC CLOSE have been changed to CIRCULAR ARC 3 POINT and CIRCULAR ARC 3 POINT CLOSE to distinguish these elements from the new elements.

Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 01-003

Comment Topic: Formal Specification

Comment: The formal grammar does not permit picture prefix elements in the DEFAULTS REPLACEMENT

Action: Accepted

Response: The <picture descriptor element> has been added to the definition of <element default> in Part 1 Section A.1.1.

Comment Number: 01-004, 01-005

Comment Topic: Metafile Descriptor

Comment: A default can be specified which does not match the mode selected (or defaulted) for a picture. What is the interaction between the "modes" and the existence of settable defaults?

Action: Clarification added to the document

Response: The following paragraphs were added to Part 1 Section 5.2.11.

"The parameters in the defaults replacement list are order dependent. When an element is encountered in the defaults replacement list, the value replaces the current default value for the element. If an element occurs more than once in the defaults replacement list, then the last value specified is the default value used by BEGIN PICTURE.

The default values for some attribute lements are implicitly associated with a specification mode. When the value for one of these attributes is set in the defaults replacement list, it must be a legal value in the 'current mode' of the attribute. 'Current mode' means either the default specification mode for the attribute or the last mode specified in the defaults replacement list. Hence, multiple mode elements may have multiple default values, one for each of the specification modes. The default replacement list may replace the Chapter 6 default values for one or more of the default values associated with the specification modes."

Comment Number: 01-006

Comment Topic: Graphical Output

Comment: It is inconsistent to permit SET ASF between TEXT and APPEND TEXT elements, but prohibit other non-text attribute changes.

Action: No change to the document

Response: SET ASF is not allowed between TEXT and APPEND TEXT elements.

Comment Number: 01-007

Comment Topic: Metafile Descriptor

Comment: Does the VDM ELEMENT LIST have to include the opcodes for itself, BEGIN METAFILE, and END METAFILE in order for the metafile to be conforming?

Action: Change to the document

Response: It is legal for these opcodes to be omitted or included since the first two have been scanned by the time the VDM ELEMENT LIST occurs and the last element must be present for the metafile to be legal. It is only necessary to include elements in the VDM ELEMENTS LIST if those elements are not required for metafile conformance.

The description of VDM ELEMENT LIST (Secti5.2.10) has been modified to read:

"All of the elements that may be encountered in the metafile and that are not mandatory are listed."

Comment Number: 01-008, 07-004

Comment Topic: Attribute

Comment: Make the starting values for index elements consistent.

Action: Change to the document, but not exactly as requested

Response: We feel that the existence of a standardized set of values in GKS should be the overriding consideration. While internal consistency would be desirable, we feel that deviation from the first and (currently only) higher level standard would be arbitrary and not justified on this point.

Comment Number: 01-009

Comment Topic: Graphical Output

Comment: Add an DISJOINT POLYLINE element to the VDM.

Action: Accepted

Response: The element DISJOINT POLYLINE has been added to the VDM. Part 1 Section 5.5.2 describes the new element DISJOINT

POLYLINE. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D, and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 01-010

Comment Topic: Graphical Output

Comment: Add a RECTANGLE element to the VDM.

Action: Accepted

Response: The new element RECTANGLE has been added to the VDM. Part 1 Section 5.5.7 describes the new element RECTANGLE. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D, and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 01-011

Comment Topic: Front Material

Comment: Add hooks to the VDM to allow adding all of the VDI graphical primitive and attributes to the VDM.

Action: No change to the document

Response: It is not the intent of the VDM to add all VDI graphical primitives and attributes.

Comment Number: 01-012, 03-002

Comment Topic: Graphical Output

Comment: Add a RESTRICTED TEXT element to the VDM.

Action: Accepted

Response: A new graphical element RESTRICTED TEXT has been aded to the VDM. Part 1 Section 5.5.14 describes the new element RESTRICTED TEXT. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D, and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 01-013

Comment Topic: Attributes

Comment: Standardize some font names

Action: Accepted

Response: Registration of font names for the VDM is accepted in principle. At the June 1984 ISO TC97/SC5/WG2 meeting provisions for a registration authority were adopted. The VDM document will be updated to refer to this registration for font names.

Comment Number: 01-014

Comment Topic: Control

Comment: Add a metafile descriptor element MAX VDC RANGE.

Action: No change to the document

Response: The suggestion only addresses one aspect of handling of high dynamic range arithmetic, and would not be valuable enough by itself to warrant the inclusion of a new element. A different approach to another part of this problem is discussed in public comment 03-015.

Comment Number: 01-015

Comment Topic: Graphical Output

Comment: The description of marker clipping is overconstrained.

Action: No change to the document

Response: The current description of marker clipping is technically correct and matches the GKS functionality.

Comment Number: 01-016

Comment Topic: Attributes

Comment: HATCH INDEX should be treated consistently with LINE TYPE and MARKER TYPE, that is, reserve the positive numbers for future standardization and negative numbers for private

types.

Action: Accepted

Response: The following paragraph has been added to Section 5.6.14.

"Non-negative values of the index are reserved for future standardization, and negative values are available for implementation-dependent use."

Comment Number: 01-017

Comment Topic: Attributes

Comment: Standardize a small number of hatch styles

Action: Accepted

Response: Six standard hatch styles have been added to the VDM. The following paragraphs were added to Section 5.6.14.

"The following hatch indices are assigned:

1: horizontal equally spaced parallel lines
2: vertical equally spaced parallel lines
3: positive slope equally spaced parallel lines
4: negative slope equally spaced parallel lines
5: horizontal/vertical crosshatch
6: positive/negative slope crosshatch

The ideal angle for the positive slope hatch patterns is +45 degrees, and the ideal angle for the negative slope hatch patterns is +135 degrees. (See Appendix D for further discussion.)"

Comment Number: 01-018, 03-020

Comment Topic: Attributes

Comment: Rename PATTERN REFERENCE POINT to FILL REFERENCE POINT.

Action: Accepted

Response: The document is changed to reflect that reference point applies to hatch as well as pattern, and the name of the element is changed to FILL REFERENCE POINT. Part 1 Section 5.6.17 has been changed to discuss the use of FILL REFERENCE POINT for both hatch and pattern.

Comment Number: 01-019

Comment Topic: Control

Comment: Add a separate LOCAL BACKGROUND COLOR element for each primitive.

Action: Changes to the document

Response: Although the technical problems are acknowledged it was felt that there were other similar problems with bundled and individual attributes. The problem is not as severe as the comment implies, mixing of bundled and individual does work, albeit implicitly. The proposed solutions have their own set of problems.

Part 1 Section 5.3.6 has been modified to describe the element in terms of the expected effect, rather than refering to specific hardware features. Recommendations have been added Part 1 Appendix D.

Comment Number: 02-001

Comment Topic: Attributes

Comment: Separate perimeter visibility from the INTERIOR STYLE element.

Action: Accepted

Response: A new element, PERIMETER VISIBILITY, has been added to the VDM. Part 1 Section 5.6.12 describes the new element PERIMETER VISIBILITY. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D, and Part 2, Part 3 and Part 4 have been updated to reflect the changes.

Comment Number: 03-001

Comment Topic: Encodings

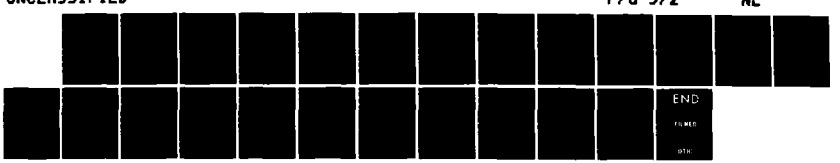
Comment: Remove the binary encoding from the VDM or make specific formats for binary data items system dependent.

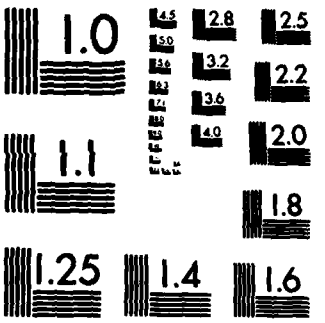
Action: Add section on conformance to Chapter 1

Response: Though at present there is no standard or de facto standard word size or binary data format across computer architectures, there is a demand for a standardized binary

AD-A153 322 STUDY OF RASTER METAFILE FORMATS(U) BATTELLE COLUMBUS LABS OH M R TAYLOR ET AL. JAN 85 ETL-0363 DAAG29-81-D-0100 3/3

UNCLASSIFIED F/G 9/2 NL





encoding of the VDM in order to promote machine-independent interchange of binary files. The suggestion to remove Part 3, VDM Binary Encoding, from the draft standard conflicts with the timely meeting of this requirement. The alternative presented here, that of defining a "standard" binary encoding with system-dependent data formats does not promote intersystem portability.

Binary encodings similar to the one defined in the VDM draft have been used across computer architectures without noticable degradation in CPU performance. If an additional increase in CPU efficiency is required, particularly when binary interchange occurs between systems with the same architecture, it is suggested that a private binary encoding based on an internal data format be used.

A new section discussing conformance of bindings (Part 1 Section 1.4.3) has been added to the document. Appendix D has been expanded to provide suggested minimum criteria for private bindings. The conformance statement is as follows:

"A functionally conforming metafile may use a private binding. While it is beyond the scope of this standard to standardize rules for private bindings, Appendix D suggests minimum criteria which private bindings should meet."

Comment Number: 03-003

Comment Topic: Attributes

Comment: Add elements to control pixel drawing shape and line ndpoint conditions.

Action: No change to the document

Response: This issue was discussed at the Timberline meeting (May 1983). The committee feels the proposal goes too far beyond the goal of standardization of a "minimal useful set of elements." The suggestion goes too far in specifying a parallel alternative method of line width control.

Comment Number: 03-004

Comment Topic: Attributes

Comment: Augment the discussion of bundles to allow interpreters to use any available device attributes to guarantee

distinguishable bundles.

Action: No change to the document

Response: The committee feels that the attribute and bundle model of the VDM is well defined and provides basic useful functionality. It also is a clear structural model that anticipates future standardization of SET <xxx> BUNDLE REPRESENTATION, where distinguishability is but one purpose of bundles (and would allow one to do the same now with the ESCAPE element). The committee voted to leave the model as is and to leave the Appendix D statement as is.

Comment Number: 03-005

Comment Topic: Front material

Comment: Remove the reference to the "presentation level" from the Abstract.

Action: No change to the document

Response: The intent of the statement was to reference the Open Systems Interconnection and to indicate the relationship of the VDM to that model.

Comment Number: 03-006, 03-008, 03-009, 03-012, 03-014, 03-017, 03-019, 03-021, 03-025, 03-027, 07-013

Comment Topic: General

Comment: General editorial changes

Action: Make changes to the document

Response: Thanks to the commentors for their editorial comments on the document. All of the suggested changes have been made in the document.

Comment Number: 03-007

Comment Topic: Front material

Comment: Clarify the definitions of "dot" and "pixel".

Action: Change to the document

Response: Dot is used intuitively in several places in the document. Its only official usage is as the name of a particular Marker type. The definition of "dot" will be removed from the glossary.

Comment Number: 03-010

Comment Topic: Attributes

Comment: Clarify the effect of changing CHARACTER HEIGHT on the existing value of character spacing?

Action: No change to the document

Response: Character spacing, the attribute, is a fraction of text height. Realized intercharacter spacing is the VDC measure obtained by applying the fraction to the height. When height changes the fraction, which is an independent attribute, does not change. The product of the two, which is the VDC measure, does change.

Comment Number: 03-011

Comment Topic: Attributes

Comment: Why is it possible to specify the expansion of color range during metafile interpretation, but not the compression?

Action: Changes to the document

Response: The text in question has been supplemented so that the symmetric mapping is specified, and moved to Appendix D, "Interpreter Guidelines".

Comment Number: 03-013

Comment Topic: Metafile Descriptor

Comment: Either reference a more specific registration authority for fonts and typefaces or only list private names as examples of font list entries.

Action: Change to the document

Response: Registration of font names for the VDM is accepted in principle. At the June 1984 ISO TC97/SC5/WG2 meeting provisions for a registration authority were adopted. The

VDM document will be updated to refer to this registration for font names.

Comment Number: 03-015

Comment Topic: Control

Comment: Need additional wording to clarify the intent of an imposable coordinate space in the VDM.

Action: Change the document

Response: VDC EXTENT is now a picture description element (see Section 5.4.6). Most of the points of the suggested additional paragraph are already adequately covered in the present wording. The sentence "It should be noted that the used of VDC EXTENT to directly encode world coordinates of large amic range and very small granularity will likely result in performance penalties at metafile interpretation time, and may result in decreased portability if such VDC extents exceed that compatible with less robust (but still conforming) metafile interpreters." has been added to Part 1 Section 4.4.5 (VDM Tailoring).

Comment Number: 03-016

Comment Topic: Control

Comment: The specification of CLIP RECTANGLE does not avoid the possibility of implied inversion.

Action: Change the document

Response: The description of CLIP RECTANGLE has been clarified. The following paragraph was added to Part 1 Section 5.3.9.

"Interpretation of this element does not cause any inversion or change of orientation of the picture. The normal condition is that xmax > xmin and ymax > ymin. If either of these conditions are not met, the interpretation of this element is implementation-dependent. A reccommendation is provided in Appendix D."

The rest of the discussion remains unchanged.

In Appendix D.3 the following sentence has been added:

"It is suggested that any error in the parameter list of

CLIP RECTANGLE causes that element to be ignored."

Comment Number: 03-018

Comment Topic: Graphical output

Comment: Clarify the effect of control characters in text strings

Action: Change the document

Response: The paragraph in Part 1 Section 4.6.6 describing the selection of characters and the use of control characters in text strings has been replaced by the following paragraph:

Select on of characters from different character sets within a string may be done in several ways, which are selected with the CHARACTER CODING ANNOUNCER metafile descriptor element. The default or normal technique is to use the CHARACTER SET INDEX element, and restrict the contents of the text strings to printing characters and spaces (format effector control codes such as CR and LF are permitted, but their interpretation is implementation-dependent). Other settings of the CHARACTER CODING ANNOUNCER or use of the ALTERNATE CHARACTER SET INDEX permit standardize use of 8-bit characters and the SI, SO and ESC control codes within the text string, in accordance with ANSI X3.41 and ISO 2022. The ALTERNATE CHARACTER SET INDEX element is used to select a character set to be used as the G1 set. This G1 set is used both for 8-bit characters in columns 10-15 of the code table, and with the SO control code. The assignment or meaning to the index parameter of both CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX is cone with the CHARACTER SET LIST metafile descriptor element."

Part 1 Section 5.5.12, Section 5.5.13, and Section 5.5.14 have been modified to discuss the effect of control characters in text strings.

Comment Number: 03-022

Comment Topic: Attribute

Comment: Add to the description of TEXT FONT INDEX that the font table is built from the font list specified in the metafile descriptor.

Action: No change to the document

Response: The current discussion section states this.

Comment Number: 03-023

Comment Topic: Attributes

Comment: The scaling of character spacing introduced in the last paragraph of the description of CHARACTER SPACING should be expanded upon in section 4.6.6.

Action: Change to the document

Response: The following paragraph has been added to Part 1 Section 4.6.6:

"The ratio of the length of the width vector to the length of the height vector is used to scale the CHARACTER SPACING for character paths, RIGHT and LEFT, and the CHARACTER EXPANSION FACTOR in all cases, before these are used to display the text."

Comment Number: 03-024

Comment Topic: Control

Comment: Clarify the default range and granularity of VDC EXTENT in section 6.1.

Action: Change to the document

Response: Part 1 Section 6.1 has been changes to indicate the default range of the VDC extent is 0.0 to 0.99999... . The granularity of the VDC extent is implied by the eding.

Comment Number: 03-026

Comment Topic: Appendix D

Comment: Problems and inconsistencies with Appendix D.

Action: Changes to the document

Response: Several changes have been made to Part 1 Appendix D. Section D.8 new lists minimum suggested interpreter capabilities and Section D.9 contains guidelines for design

of private bindings.

The clipping of graphical elements discussion in Section D.3 has been clarified. The former GKS model for color to grey scale mapping has been deleted.

Comment Number: 04-001

Comment Topic: Attributes

Comment: The marker mode of scaled can be interpreted to produce transformable markers. Add prose to the standard to specify the interpretation. Annotation text should be supported by the VDM in the same manner.

Action: No change to the document

Response: Your interpretation of scaled marker mode as capable of producing 'transformable' markers is a valid interpretation, but not the primary motivation of the mode. The motivation is to allow different relative sizes to be selected without having to take the absolute size in VDC space into consideration.

The VDM was designed to serve multiple device and applications models. Adoption of your suggestion would constrain the meaning to apply to one particular model.

Annotation text can be supported by setting a constant VDC EXTENT, scaling text appropriately into that range, and using driving software to scale graphics output. This is consistent with the view that VDC EXTENT is intended for use in device tailoring, not as world or windowing coordinates with which to build a viewing system.

Comment Number: 04-002

Comment Topic: Global

Comment: While it is impossible to foresee all possible extensions required in the future, I seek some reassurance that the committee developing the VDM standard consider this proposal easily extensible and extendable.

Action: No change to the document

Response: At the June, 1984 ISO TC97/SC5/WG2 Metafile Subgroup

meeting a careful study of the VDM document was made to ensure that nothing in the document would preclude future extensions.

Comment Number: 04-003

Comment Topic: Global

Comment: Add elements for segmentation to the VDM.

Action: No change to the document

Response: X3J6 and X3V1 have separate standards for editing and formatting data. Segmentation does not fit with the existing functionality.

Segmentation is beyond the stated scope of the standard, as has been re-affirmed repeatedly by the task group. See Part 1 Section 1.2, item 3.

Comment Number: 04-004

Comment Topic: Global

Comment: Add 3D elements to the VDM

Action: No change to the document

Response: The VDM is a picture transfer system not a model capture system. Three dimensions is beyond the stated scope of the standard, as has been re-affirmed repeatedly by the task group.

Comment Number: 05-001

Comment Topic: Front material

Comment: Provide for the existence of a Registration Authority to register ESCAPEs, attribute indices and the FONT LIST.

Action: Accepted

Response: Registration of font names, ESCAPEs, and attribute indices for the VDM is accepted in principle. At the June 1984 ISO TC97/SC5/WG2 meeting provisions for a registration

authority were adopted. The VDM document will be updated to refer to this registration authority.

Comment Number: 05-002

Comment Topic: Global

Comment: Add two new values to the VDM ELEMENT LIST, "MINIMAL" and "FULL".

Action: Accepted

Response: The elements of the VDM will be partitioned into two sets: the "DRAWING SET" and the "DRAWING SET PLUS CONTROL." The sets are shorthand names for sets of VDM elements. They should not be considered MACRO names, nor should they be construed to be levels of conformance. Part 1 Section 5.2.10 lists the elements contained in each of the sets and the use of the shorthand namesPart 1 Chapter 4 and Appendix A, and Part 2, Part 3 and Part 4 have been updated to reflect this change.

Comment Number: 05-003

Comment Topic: Attributes

Comment: The definition of the boundary of the fill area (centered on the ideal boundary) is not consistent with the current PHIGS draft.

Action: No change to the document

Response: The issue was discussed at the Timberline meeting. The committee position is that the current wording has the fewest undesirable characteristics, is the most implementable, and is, in principle, no more badly behaved than other alternatives given that polygons can have perimeter on or off, can abut, etc.

Comment Number: 05-004

Comment Topic: ESCAPE

Comment: The statement in section 5.1 regarding values reserved for future standardization applies also to the VDM ESCAPE function identifier parameter.

Action: Accepted

Response: The discussion in Part 1 Section 5.7.1 (VDM ESCAPE) has been modified such that non-negative values of the function identifier parameter are reserved for possible future standardization and negative values are for private use.

Comment Number: 05-005

Comment Topic: Metafile Descriptor

Comment: VDM VERSION should not have a default, because it is a required element.

Action: Change the document

Response: The default for VDM VERSION has been changed to N/A in Part 1 Chapter 6.

Comment Number: 05-006

Comment Topic: Metafile descriptor

Comment: Are the elements described in paragraphs 5.2.4 through 5.2.9 and 5.3.7 and 5.3.8 required elements? If not, section 6.1 should state that the defaults for these values are dependent upon the encoding.

Action: Change to document

Response: All of the elements referenced are precision setting elements, except for MAXIMUM COLOR INDEX. Precision setting elements have binding dependent formats and defaults. Part 1 Chapter 6 has been changed to indicate the default for these elements is binding dependent.

MAXIMUM COLOR INDEX has a fixed format, but has binding dependent defaults. MAXIMUM COLOR INDEX has been added to Part 1 Chapter 6.

Comment Number: 05-007

Comment Topic: Attributes

Comment: The formal definition of TEXT ALIGNMENT shows the continuous align value as optional, but the binary binding shows these elements as required.

Action: Change the document

Response: The optionality has been removed from the formal specification and the bindings.

Comment Number: 05-008

Comment Topic: Appendix D

Comment: The guidelines for TEXT ALIGNMENT should not recommend any fallback; rather, all metafile readers should do the best they can with the text facilities available in the supported device.

Action: Change to the document

Response: The discussion of TEXT ALIGNMENT in Part 1 Section D.5 has been modified as follows:

"If an alignment value is not available, the closest available value is used."

Comment Number: 05-009

Comment Topic: Appendix D

Comment: ARC and ARC CLOSE elements with only two distinct coordinates should not be ignored, but should appear as a line.

Action: Change to the document

Response: The following sentence has been added to the discussion of ARC in Part 1 Section D.6:

"If an ARC element has only two distinct points a line is drawn between the points."

The following sentence has been added to the discussion of ARC CLOSE in Part 1 Section D.6:

"If an ARC CLOSE element has only two distinct points a line is drawn between the points."

Comment Number: 05-010

Comment Topic: Appendix D

Comment: Delete the last paragraph of D.7. This algorithm is no longer in GKS.

Action: Change to the document

Response: The format GKS algorithm for mapping color to grey scales has been deleted from Part 1 Section D.7.

Comment Number: 05-011

Comment Topic: Character encoding

Comment: Maxm color index for the character coded binding is missing from the list in Part II section 9.

Action: Change to the document

Response: A MAXIMUM COLOR INDEX default of 63 will be adopted.

Comment Number: 05-012

Comment Topic: Binary encoding

Comment: The default for MAXIMUM COLOR INDEX should not be 0; rather a value (e.g. 16) should be stated for the case that color indices are used.

Action: Accepted

Response: The value 63 has been adopted as the default for MAXIMUM COLOR INDEX.

Comment Number: 05-013

Comment Topic: Binary encoding

Comment: Maximum color index for the binary binding is missing from the list in section 14.

Action: Change to the document

Response: The comment is accepted. The value 63 has been adopted as the default.

Comment Number: 06-001

Comment Topic: Bindings

Comment: It is very important to maintain consistency across the different bindings as to whether a functions is specified by an opcode or a parameter.

Action: Change to the document

Response: A new section D.9 "Guidelines for Private Bindings" has been added to Part 1. The new section recommends:

"All VDM elements must have a specified encoding, with the exception of the precision commands, which may not be applicable to a particular binding. An element which sets an interpretation mode for other elements may be implicit in the commands which it affects, as opposed to being coded as a separate element."

The following paragraph was removed from Part 1 Section 5.1.

"The order in which parameters will occur in a parameter list is not to be assumed from the order in which they are mentioned in this section, but is deferred to the description of specific encodings."

The order of parameters in Part 1 Section 5 and Appendix A, and Part 2, Part 3, and Part 4 is now consistent.

Comment Number: 06-002

Comment Topic: Bindings

Comment: Make the mapping from the list of enumerated values to integers the same across all bindings which use integers for enumerated types.

Action: Accepted

Response: All of the bindings now use the values from the binary binding except for the final flag for TEXT and APPEND TEXT. In this case use 0 for not final and 1 for final.

Comment Number: 06-003

Comment Topic: Character encoding

Comment: Several minor mistakes and areas where clarification is required.

Action: Accept

Response: The suggested changes were made to Part 2.

Comment Number: 07-001

Comment Topic: Graphical Output

Comment: Does the addition of the POLYGON SET element remove the need for handling of complex polygons.

Action: No change to the document

Response: We do not believe that the addition of POLYGON SET reduces the need for complex polygon handling. Detection of complex polygons is non-trivial and therefore their prohibition is expensive; their execution adds very little complexity if POLYGON SET is implemented.

Comment Number: 07-002

Comment Topic: Attributes

Comment: Allow line type and perimeter type to restart a pattern at each vertex.

Action: No change to the document

Response: We are not standardizing the interpreter nor saying whether implementations are conforming or not. We are defining what comprises a syntactically correct metafile and defining what we feel are the proper semantics of elements. Because polyline is a single graphical element and the nodes are simply part of its defining syntax, linestyle should be continuous.

Comment Number: 07-003

Comment Topic: Graphical output

Comment: An implementation should be allowed to draw two lines for a three point arc when the calculation of the radius and center point would require multiple precision arithmetic.

Action: No change to the document

Response: This should not be addressed by the standard. It is an implementation degeneracy handling decision.

Comment Number: 07-005

Comment Topic: Control

Comment: The control element VDC EXTENT should be a picture descriptor element.

Action: Accept

Response: The element VDC EXTENT has been moved to Part 1 Section 5..6. Part 1 Chapter 4 and Appendix A have been updated to reflect this change.

Comment Number: 07-006

Comment Topic: Formal grammar

Comment: Table A-1 and figure A-1 fail to incorporate the control elements.

Action: No change to the document

Response: Control elements are contained in the definition of PICTURE ELEMENT.

Comment Number: 07-007

Comment Topic: Appendix D

Comment: The Minimum Required Capability list indicates that the CHARACTER SET INDEX minimum is two, but the CHARACTER SET LIST minimum is one. Please clarify.

Action: Change to the document

Response: The character set index minimum has been changed to 1.

Comment Number: 07-008

Comment Topic: Character encoding

Comment: In Part II section 7.2 no description is given for the format of the exponent part of real numbers.

Action: Change to the document

Response: A reference to Section 7.1 has been added after the last sentence of Section 7.2.

Comment Number: 07-009

Comment Topic: Graphical output

Comment: The specification of the ARC element does not indicate if there is an assumed sweep direction for the three points defining the arc.

Action: No change to the document

Response: We do not understand the need for specifying the sweep direction, feeling that this is an implementation issue. The arc is uniquely defined without the additional information unlike the center-radius-angle specifications.

Comment Number: 07-010

Comment Topic: Appendix D

Comment: The guidelines for CHARACTER ORIENTATION make use of the term "counterclockwise". Is the intent that the "counterclockwise" direction be take irrespective of the sense and orientation of the VDC EXTENT?

Action: Change to the document

Response: The discussion of CHARACTER ORIENTATION in Part 1 Section D.5 has been changed as follows:

"If two are equally near, the one in a positive angular direction is chosen."

Comment Number: 07-011

Comment Topic: Global

Comment: Add an appendix with a table of implementation dependencies.

Action: No change to the document

Response: Given the wide range of interpreters and their expected capabilities and limitations, reliably identifying such dependencies is felt to be an impractical, and would probably lead to misleading results. Such items are dealt with as much as possible in Appendix D.

Comment Number: 07-012

Comment Topic: Binary encoding

Comment: In Part III tables 3-8 should provide relevant information for the parameters of all the elements in that group.

Action: Accept

Response: Tables 3-8 updated.

Comment Number: 08-001

Comment Topic: Front material

Comment: The relationship of the VDM standard to ANSI X3.110 (NAPLPS) should be stated.

Action: Change to the document

Response: The following description was moved from Part 1 Section 2.2 to Part 1 Section 2.1.

"While there are similarities between the VDM and the North American Presentation-Level Protocol Syntax (NAPLPS: ANSI X3.110-1983), the latter is designed to support a particular class of devices in a picture transmission environment, while the VDM is intended to provide picture definition in a device-independent and environment-independent manner."

Comment Number: 08-002, 08-007

Comment Topic: Attributes

Comment: The description of color does not reference any standard.

Action: No change to the document

Response: The VDM provides a mechanism for recording and transmitting Red, Green, Blue color components. It purposely does not address questions such as this. Your points are well taken, but favor one technology ovothers. The VDM thus leaves the exact meaning of the components to the applications. Note the existence of APPLICATION DATA allows the documentation of special meanings, and VDM ESCAPE allows sending device-dependent commands relating to color interpretation.

Comment Number: 08-003

Comment Topic: Character encoding

Comment: In the character-coded binding for NON VDC REAL PRECISION and REAL VDC PRECISION, the intent of the smallest-real -code seems to be to indicate the number of exponent bits, but this is not stated.

Action: Change to the document

Response: Part II Sections 8.5 and 8.22 have been reworded to incorporate your comments.

Comment Number: 08-004, 11-004

Comment Topic: Character encoding

Comment: Delete all references to "G-set functions" and the "VDM G-set." Indicate that the VDM coding environment may be invoked (a) implicitly, or (b) from an ISO 2022 coding environment by means of an ESC 2/5 F sequence to be assigned by the Registration Authority of ISO 4873. Also, clarify that the concepts of "C0 sets," "C1 sets," and "G-sets" apply only within the string parameters of those metafile elements which have string parameters.

Action: Accept

Response: The changes you suggested have been made to Part II Section

1, Section 2.2, and Section 3 of the document.

Comment Number: 08-005a, 12-001

Comment Topic: Character encoding

Comment: Since the VDM coding environment is invoked from ISO 2022 as "another coding system", there is no need to describe bit combinations from columns 0 and 1 of the VDM code chart as "control characters."

Action: Accept

Response: Throughout Part II of the document "IS2" has been replaced by "MET" (METAFILE ELEMENT TERMINATOR) and "IS1" has been replaced by "MPT" (METAFILE PARAMETER TERMINATOR).

Comment Number: 08-005b

Comment Topic: Character encoding

Comment: There is no provision for issuing commands (metafile elements) in which a parameter other that the last parameter is omitted. It may be desired, at least in future versions of this standard, to provide for omitting parameters.

Action: Accept

Response: Part II Section 7.7.3 has been modified to include your comment. MET (Metafile Element Terminator) is now 1/13, MPT (Metafile Parameter Terminator is now 1/15, and 1/14 is reserved for future standardization.

Comment Number: 08-006, 11-005

Comment Topic: Attributes

Comment: Add a mechanism to invoke 8-bit graphic character sets as well as 7-bit graphic character sets.

Action: Accept

Response: Part 1 Section 5.6.23 describes the new element ALTERNATE CHARACTER SET INDEX. Part 1 Chapter 4, Chapter 5, Chapter 6, and Appendix A , and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 08-008, 10-001, 13-001, 13-003

Comment Topic: Character encoding

Comment: Use a coding system which uses the ISO 2022 character set structure and control code structure rather than a separate coding system.

Action: Reject

Response: We accept the majority ANSC X3L2 opinion and the decision of ISO TC97/SC2 to handle the Character-Coded Binding of VDM as a complete code, entered and exited with escape sequences registered with the Registration Authority of ISO 4873. We concur with the ISO character coding committees in the view that the graphics environment and user model is sufficiently different from that of ISO 2022 and 6429 that attempting to force both to be handled with the same protocol would compromise both standards.

We cannot accept your suggestion to use ISO 6429 controls in a way which is similar to but different from ISO 6429. It is inappropriate to base a standard on a proposed future update of another standard. To the extent that dpANS VDM and ISO 6429 handle similar functions, we expect that the pbest solution will be worked out in the marketplace and incorporated into future versions of both standards.

[illegible] single coding environment.

Similarly, it is not necessary to have all standards use the same coding environment in order to accomplish our objective of a consistent set of standards. The complete code technique enables us to combine different standards in an orderly fashion, while permitting each standard to address the needs of its primary constituency in the most appropriate manner.

Comment Number: 08-009, 13-002

Comment Topic: Character encoding

Comment: Code the VDM as ISO 6429-style controls strings rather than a complete code and use the ISO 2022 mechanism for accessing a G1 set rather than "ALTERNATE CHARACTER SET INDEX".

Action: Reject

Response: We accept the majority ANSC X3L2 opinion to add the ALTERNATE CHARACTER SET INDEX element. This is compatable with a previous resolution to use CHARACTER SET INDEX rather than the ISo 2022 mechanism to select the G0 set. Most graphics implementations do not handle the ISO 2022 mechanism of designation and invocation of character sets, thus the dpANS VDM reflects current practice in the graphics industry.

A CHARACTER CODING ANNOUNCER element has been added to the metafile descriptor, which provides a standardized way of recording the writer's intent to use the full 7-bit or 8-bit ISO 2022 functionality within text strings. Thus this technique of character set selection is available for those who prefer it over the one standardized in the dpANS VDM. Use of this functionality is by prior agreement between the writer and interpreter of the metafile, however, and is not required to be supported or used.

Comment Number: 11-001

Comment Topic: Front material

Comment: In Part I paragraph 2.3 which GKS is meant (ISO or ANSI)

Action: Change to the document

Response: The reference to GKs in Part I Section 2.2 now reads "the Graphical Kernel System (GKS: BSR dp ANS X3.124-198x) and the reference in Section 2.3 now reads "The Graphical Kernel System (GKS: ISO DIS 7942)".

Comment Number: 11-002

Comment Topic: VDM ESCAPE

Comment: Add a GDP element, separate from the VDM ESCAPE element, for private geometrical outputs.

Action: Change to the document

Response: WG2 has added a GKS-compatible GDP to the ISO version of the VDM. The ANSI VDM will include it as well. (NOTE: These changes are not incorporated into the August, 1984 version of the document.)

Comment Number: 11-003

Comment Topic: Attributes

Comment: The VDM provides, in addition to the FILL AREA attributes of GKS, specific attributes related to the perimeter and a perimeter visibility flag. To be a coding of ISO GKSM, the VDM should either handle the POLYGON element as a GDP or delete from VDM the elements and parameters related to the perimeter.

Action: No change to the document

Response: The additional functionality that you note was added in response to requests from and to serve the needs of a non-GKS constituency. We have attempted to ensure that the needs of the GKS constituency are not compromised while adding functions to serve other groups.

Comment Number: 12-002

Comment Topic: Character encoding

Comment: The new proposed text from X3L2 for section 7.7.3.2 does not allow any of the commands currently in the VDM standard besides TEXT and APPEND TEXT to ever be revised to give meaning to the use of the 8th bit.

Action: Accept

Response: Part 2 Section 7.7.4.2 has been revised as follows.

"This standard does not specify the effect, if any, of using character from code chart columns 10 through 15 in the string parameters of metafile elements other than TEXT, APPEND TEXT, and RESTRICTED TEXT, nor will it in any future revision of this standard.

Comment Number: 14-001

Comment Topic: Binary encoding

Comment: Allow two forms of CELL ARRAY in the binary binding: one where pixels are aligned according to the COLOR INDEX PRECISION; and another where pixels are represented in the minimum number of bits as derived from the MAXIMUM COLOR INDEX metafile descriptor element.

Action: Change to the document

Response: The cell array in the binary encoding (Part 3) has been modified to include an additional precision parameter with values of 1, 2, 4, 8, 16, or 24 bits of pixel precision. Each row starts on a VDM word boundary.

Comment Number: 15-001

Comment Topic: Metafile Descriptor

Comment: Add a new element CHARACTER CODING ANNOUNCER which identifies the code extension technique and environment assumed by the generator of the metafile.

Action: Accept

Response: The new element CHARACTER CODING ANNOUNCER has been added to the VDM. Part 1 Section 5.2.13 describes the new element CHARACTER CODING ANNOUNCER. Part 1 Chapter 4, Chapter 5, Chapter 6, Appendix A and Appendix D, and Part 2, Part 3, and Part 4 have been updated to reflect the changes.

Comment Number: 15-002

Comment Topic: Metafile descriptor

Comment: There are some inconsistencies with the CHARACTER SET LIST element.

Action: Changes to the document

Response: The description of CHARACTER SET LIST in Part 1 Section 5.2.14 has been modified to incorporate your suggestions.

END

FILMED

5-85

DTIC